UNIVERSIDADE DE BRASILIA
FACULDADE DE TECNOLOGIA
DEPARTAMENTO DE ENGENHARIA ELÉTRICA

TESE DE DOUTORADO EM ENGENHARIA ELÉTRICA

CRIPTOANÁLISE DE PROTOCOLOS DIRECIONADOS A DISPOSITIVOS DE BAIXO PODER COMPUTACIONAL

José Antônio Carrijo Barbosa

Tese submetida ao Departamento de Engenharia Elétrica como requisito parcial para obtenção do grau de Doutor em Engenharia Elétrica

Banca Examinadora

Prof. Anderson Clayton Alves Nascimento, _____
ENE/UnB - Dr., Orientador

Prof. Francisco Assis de Oliveira Nascimento, _____
ENE/UnB - Dr., Examinador interno

Prof. Júlio César López Hérnandez, _____
IC/UNICAMP - Dr., Examinador externo

Prof. Rafael Timóteo de Sousa Júnior, _____
ENE/UnB - Dr., Examinador interno

Prof. Ricardo Dahab, _____
IC/UNICAMP - Dr., Examinador externo

Dedicatória

Dedico estes resultados ao meu pai e a minha mãe, eles foram únicos.

José Antônio Carrijo Barbosa

Agradecimentos

Agradeço ao CEPESC pelos anos trabalhados exclusivamente em criptografia, ao Departamento de Engenharia Elétrica pelo acolhimento ao meu projeto, e ao prof. Adson pela orientação inicial e a forma tão receptiva que me foi dada. Ao prof. Anderson que tive e tenho a sorte de conhecê-lo e tê-lo como orientador e amigo, graças à sua orientação, dedicação, confiança e conhecimento técnico me sugeriu cada rumo da pesquisa aqui apresentada: Anderson, você é tudo de bom. Ao Rafael Tonicelli pela colaboração constante. Ao meu pai, só orgulho de tê-lo tido; a minha mãe, o meu amor de filho simplesmente grato, tenho saudades... Aos meus quatro irmãos Antônio Filho, Astério, Getúlio e Clarêt, e as duas irmãs Esperança e Pia, pela paz que sempre reinou entre nós. Aos meus cunhados e cunhadas, sobrinhos e sobrinhas e suas esposas e maridos, vocês são a nossa continuidade. A minha esposa Tutta que me apoiou cada dia em todo este projeto, toma conta da minha vida e me deu quatro filhos maravilhosos, te amo; aos meus filhos Tobias, Tomáz, Hugo e Tuzza que me incentivaram estudando junto comigo e me cobraram resultados, sou muito feliz por ter todos vocês.

José Antônio Carrijo Barbosa

RESUMO

O uso de dispositivos que geram comunicação sem contato, possibilitando identificação por rádio frequência, como os RFID (*Radio Frequency Identification*), tem avançado bastante em todo o mundo por permitirem transações rápidas, serem duráveis, de fácil manuseio e seu preço estar em queda. Por estes motivos o RFID, que é utilizado principalmente como um meio para identificação, aos poucos tem substituído sistemas como o de Código de Barras. Além disso, a tecnologia RFID está sendo utilizada também para validação dos mais diversos documentos como passaportes e passes pessoais, para rastreamento de carnes e de produtos oriundos das indústrias em geral.

Este mercado em ascensão tem gerado a necessidade de criação e análise de algoritmos e de protocolos criptográficos direcionados a estes dispositivos, notórios pelo seu baixo poder computacional. Vários protocolos têm sido criados, porém muito poucos diferentes métodos de ataques têm sido propostos. Com esta preocupação quanto a importância destes sistemas e também com a privacidade dos usuários, nós analisamos e criamos várias novas técnicas de ataques a alguns desses protocolos e algoritmos direcionados a dispositivos de baixo poder de processamento.

Este trabalho apresenta nove resultados referentes a ataques criptoanalíticos que criamos para protocolos direcionados a esses dispositivos. Um dos resultados corresponde a um ataque passivo e probabilístico ao protocolo HB, extensivo aos protocolos HB^{+}, HB^{++}, $HB^{\#}$s e HB*, onde obtivemos resultados bastante eficientes, na maioria dos casos melhores do que todos aqueles já apresentados na literatura, como o método BKW [12] e o método otimizado de Fossorier et al. [22]. Uma outra linha de nossa pesquisa que desenvolvemos e obtivemos resultados bastante expressivos foi a de criação de métodos de criptoanálise utilizando-se de ataques por falha aos protocolos HB, HB^{+}, $HB^{\#}$ e Random-$HB^{\#}$, sendo extensivos ao protocolos HB^{++} e HB*. Nós também desenvolvemos linhas de pesquisa referentes a criação de métodos de ataques, um ativo e um passivo, a protocolos baseados no problema da soma de k mínimos, assim como criamos métodos de ataques, um ativo e um passivo, ao protocolo HB-MP. Por fim, criamos também um método de ataque ativo ao algoritmo simétrico baseado em autômata celular, regra 30, possibilitando a recuperação de toda a chave utilizada.

ABSTRACT

Radio Frequency Identification Devices (RFID) are gradually becoming a common part of our daily lives. Initially thought as an improved substitute to bar-code based identification systems, they are now being used for identifying valid passports, counterfeit gods and for tracking animals.

RFID based systems are mostly used for providing efficient identification. It is, thus, essential to ensure that they are resistant against attacks performed by malicious users attempting to impersonate a legal user. This task, known as the identification problem, is well known in cryptography. However, the classical solutions usually cannot be applied to the RFID environment, due to the lack of computational power and restrictions to bandwidth and battery consumption. Thus, one

has to design solutions suitable to this tightly restricted environment and analyze their proposed security .

This thesis analyzes several cryptographic protocols (mostly identification protocols) proposed for computational devices with low computational power. More specifically, we design several novel attacks against the family of protocols HB, against identification protocols based on the hardness of the sum of k mins problems and against a stream cipher based on cellular automata. Our results significantly improve over previous published attacks. Moreover, we introduce, to the best of our knowledge, the first fault based attacks againt RFID identification protocols.

SUMÁRIO

Capítulo 1

Introdução

O tema deste trabalho de pesquisa é a criação de novos métodos criptoanalíticos direcionados para protocolos de identificação projetados para dispositivos de baixo poder computacional. Enfoque especial é dado aos protocolos direcionados a dispositivos de identificação por rádio frequência – RFID.

1.1 Motivação

Os dispositivos de identificação por rádio frequência são etiquetas munidas de certo poder de processamento que se comunicam com uma leitora, usualmente ligada a uma base de dados. Este tipo de sistema hoje é utilizado para garantir a identidade do portador em passaportes digitais, identidades digitais, para garantir a autenticidade de ingressos para eventos, facilitar o rastreamento de animais, etc. Obviamente, segurança desponta como um quesito essencial para um uso mais amplo de tais dispositivos. No entanto, algumas dificuldades surgem no desenho de soluções de segurança para sistemas RFID (e sistemas de baixo poder computacional de maneira geral). Estes dispositivos devem ser, usualmente, baratos de modo a permitir sua utilização em grande escala. Esta limitação no preço do dispositivo implica em severas restrições no seu poder computacional, capacidade de armazenamento e velocidade de transmissão e recepção de dados.

Para as etiquetas RFID mais baratas, mesmo uma cifra de bloco ou uma função de hash podem apresentar custos proibitivos do ponto de vista de complexidade computacional. O poder de processamento de tais dispositivos é medido em quantidades de portas lógicas e normalmente não supera a figura de quatro ou cinco mil portas lógicas.

O projeto de soluções de criptografia e de protocolos de autenticação, de maneira mais particular, sob tais circunstâncias é um desafio considerável. Dentre as propostas apresentadas na literatura, a família de protocolos de autenticação HB (HB, HB^{+}, HB^{++}, HB$^{\#}$, Random-HB$^{\#}$, HB* e HB-MP) tem recebido atenção especial. Primeiro porque consiste de protocolos com custo computacional de implementação muito baixo, possibilitando a adição de segurança mesmo no caso de etiquetas RFID de custo da ordem de centavos de dólar. Segundo, por apresentar provas formais de segurança, onde o problema de quebrar o referido protocolo é reduzido a resolução de problemas considerados intratáveis.

Devido à natureza da aplicação de dispositivos RFID (como no caso de passaportes digitais), é de extrema importância que protocolos de identificação tais como os da família HB sejam analisados com cuidado. Este é o principal objetivo desta tese.

1.2 Resultados Obtidos

Listamos os principais resultados obtidos neste trabalho:

- Propomos um novo método criptoanalítico passivo contra todos os protocolos da família HB. Este é o primeiro método de ataque passivo que pode ser aplicado contra todos os membros da família. Além disso, é o primeiro ataque passivo contra os protocolos da família HB onde a quantidade de dados capturados necessários não cresce exponencialmente com o tamanho da chave. Para parâmetros práticos anteriormente sugeridos na literatura, nosso ataque quebra os protocolos da família HB.
- Ataques por falhas são ataques onde é assumido que o adversário obtém controle sobre o dispositivo usado para implementar o algoritmo criptográfico. Neste caso, assume-se também que ele é capaz de modificar regiões da memória do dispositivo, bem como reinicializá-lo quantas vezes for necessário. Propomos neste trabalho os primeiros ataques por falha contra os protocolos da familia HB. Os ataques são extremamente efetivos e práticos, evidenciando a total fragilidade de tais protocolos no caso de ataques por falhas.
- Propomos os primeiros ataques (passivo e por falhas) contra protocolos baseados na soma de k mínimos. Estes protocolos foram propostos por Blum como uma outra alternativa aos protocolos da família HB. O ataque passivo mostra-se superior a um ataque por força bruta enquanto o ataque por falhas quebra totalmente a segurança do protocolo em questão.
- Propomos um ataque por falhas contra uma cifra sequencial baseada em autômatos celulares. Este tipo de cifra também foi proposto tendo em mente seu uso em dispositivos de baixo poder computacional, dada a sua facilidade de implementação. Mostramos, no entanto, que tal facilidade vem acompanhada de um preço, a sua fragilidade contra ataques por falhas.

1.3 Organização do Documento

No capítulo 2 é apresentada uma revisão bibliográfica onde apresentamos algumas informações acerca de dispositivos RFID e os protocolos da família HB. Em seguida, no capítulo 3 descreve-se um novo ataque passivo probabilístico aos protocolos HB/HB$^+$. No capítulo 4 é apresentado um ataque por falha aos protocolos HB/HB$^+$. No Capítulo 5 é apresentado um ataque por falha a protocolos baseados no Problema da Soma de K Mínimos, enquanto que no Capítulo 6 é apresentado um ataque passivo a esses protocolos. No Capítulo 7 é apresentado uma ataque passivo e um por falha ao protocolo HB-MP. No capítulo 8 é apresentado um método de ataque por falha a protocolos baseados em autômatos celulares, regra 30. No capítulo 9 são apresentadas as conclusões dos resultados obtidos. No anexo I são apresentados alguns gráficos referentes ao desempenho

do nosso ataque probabilístico aos protocolos HB/HB$^+$. No anexo II mostramos através de gráficos a relação de desempenho entre um ataque exaustivo e o nosso ataque passivo a protocolos baseados no Problema da Soma de K Mínimos. No anexo III é apresentado em pseudo-código a implementação do nosso ataque ativo a protocolos baseados no Problema da Soma de K Mínimos.

Capítulo 2

Revisão Bibliográfica

Os protocolos criptoanalisados nesta tese são destinados aos dispositivos de baixo poder computacional, de maneira geral. No entanto, boa parte das aplicações dos mesmos na literatura aparecem no contexto de dispositivos de identificação por rádio frequência, também conhecidos pela abreviação RFID. Logo, neste capítulo, apresentamos, de maneira breve, alguns conceitos básicos acerca de sistemas de identificação baseados em RFID, seus problemas de segurança e algumas soluções propostas na literatura. Damos ênfase particular aos protocolos da família HB, principal objeto de estudo desta tese.

2.1 Tecnologia RFID

Esta sessão descreve conceitos básicos da tecnologia RFID. RFID é uma abreviação de lingua inglesa que significa *Radio Frequency Identification*. Sistemas RFID consistem, usualmente, de etiquetas providas de um certo poder computacional capazes de enviar e receber mensagens eletromagnéticas, uma leitora com antena e uma central de processamentos. Esses sistemas são normalmente usados na identificação e rastreamento de produtos e pessoas. A primeira aplicação de dispositivos RFID parece ter ocorrido no *Royal British Air force* em um sistema de identificação de "amigos", ou de "inimigos", tendo sido usado na Primeira Guerra Mundial, para identificação de aeronaves como explicam Avoine et al. [6]. Apesar de ter sido descoberta há várias décadas, a tecnologia RFID era desconhecida pela população até a pouco tempo. Com o barateamento dos chips utilizados na sua implementação, foi possivel o uso em larga escala de tais dispositivos.

De maneira informal, dispositivos RFID de baixo custo, conhecidos apenas como Etiquetas RFID, são pequenos *microchips* com aproximadamente 0.4 mm x 0.4 mm, e são tão finos que podem ser embutidos em papel conforme explicado por Takarai et al. [60]. Usados para armazenamento de dados e execução de operações lógicas, essas etiquetas não têm microprocessadores e são equipadas com apenas poucos milhares de portas lógicas. A sua memória, bastante limitada, pode ser do tipo somente leitura, escreve uma vez e lê várias vezes, ou permite várias escritas. Existem outros comandos que podem estar disponíveis nesses dispositivos como o *sleep, wake, e kill*, além dos *keyed-read e keyed-write*. O seu conceito é bastante simples, e a sua aplicação pode ser barata e efetiva. Ao chip RFID é acoplado uma antena que transmite e recebe dados a partir de questionamentos de

uma leitora de RFID, quanto maior for a antena, maior é o alcance dessa comunicação. Note que, como as respostas das etiquetas para as leitoras são de forma "involuntária", respondem sempre que questionadas, é necessário uma forma de proteção contra questionamentos não autorizados. Uma maneira de obter tal proteção no caso de etiquetas que não estão em uso é a Gaiola de Faraday, constituída de malha metálica, impenetrável por algumas frequências de onda, incluindo as utilizadas por sistemas de RFID.

A antena é uma importante parte de qualquer sistema RFID, e converte o sinal elétrico/magnético transmitido pelo ar, da leitora para a etiqueta, e vice-versa. Existem quatro diferentes alcances de leitura para etiquetas passivas que devem ser considerados, quando se discute sobre privacidade, Juels [35]: o alcance nominal, o de varredura indevida, o de monitoramente da etiqueta por uma leitora e o de monitoramento da leitora por uma etiqueta. Os RFID padronizados possuem em suas especificações as indicações do alcance nominal de leitura para que o sistema possa operar. Kfir e Wool [43] propõem que se utilize bateria com alcance máximo de leitura de até 50 cm para etiquetas passivas, onde essa é cinco vezes maior que a do padrão ISO 1444. No entanto, após a leitora ter feito o primeiro questionamento/contato com uma etiqueta, fornecendo-lhe energia, uma outra leitora não autorizada pode acessar a mesma etiqueta e começar a monitorar esta comunicação a uma distância ainda maior. O problema é que as leitoras também podem ser monitoradas por distâncias bastante superiores, até quilômetros, uma vez que essas emitem frequências mais altas.

Quanto a sua taxonomia, as etiquetas podem ser ativas, semi-passivas ou passivas, sendo estas as mais comuns. As etiquetas ativas possuem bateria, mantém o circuito interno ligado e podem começar uma comunicação; as etiquetas semi-passivas têm bateria para manter o circuito interno energizado, mas não iniciam uma comunicação; em geral, as etiquetas RFID mais baratas são passivas, operam em algumas bandas de frequência, derivam a energia para seu funcionamento a partir do sinal de questionamentos originários de uma leitora. Etiquetas com baixa frequência (LF) operam em um intervalo de 124 kHz a 135 kHz e têm um perímetro de leitura de até meio metro. Etiquetas com alta frequência (HF) operam a 13.4 MHz e têm perímetro de leitura próximo a um metro. Etiquetas com frequência ultra alta (UHF) operam entre 860 MHz a 960 MHz (em algumas vezes a 2.45 GHz), têm alcance acima de dezenas de metros, porém estão sujeitas a mais interferências do ambiente como a água, como conta Juels [35].

Embora a taxa de leitura, pela leitora, chegue a centenas de etiquetas por segundo, as leitoras não podem ler mais do que uma a cada instante. Por este motivo, em um sistema de RFID, composto por etiquetas (*microchip*+antena), leitora(s) (pode ser fixa ou móvel, e também possui antena) e servidor com banco de dados remoto, é necessária a implantação de protocolos anti-colisão, que infelizmente são frequentemente proprietários. Esses protocolos são imprescindíveis a esses sistemas porque a leitora não tem controle do destinatário para onde o sinal é enviado, então, ao enviar um sinal para uma determinada etiqueta, todas as outras que estiverem no raio de alcance dessa leitora podem interferir nessa comunicação, e iniciar também várias outras comunicações simultâneas, ocorrendo colisão.

O dispositivo RFID tem como principal função a identificação automatizada de objetos e pessoas sem contato físico ou visual, a partir de uma comunicação sem fio. Também podem ser

incorporados ao *microchip* outras funcionalidades como sensores integrados, criptografia e controle de acesso. No entanto, esses sistemas não são desenhados para prevenir-se contra ataques ou para responder a qualquer indicação de que algum ataque possa estar em progresso. Em geral, todos os ataques aos sistemas de RFID consideram a leitora e o banco de dados como uma entidade única e segura, i.e., a comunicação entre leitora e este servidor é considerada como segura, porque este detém todas as chaves do sistema.

2.1.1 Aplicações

Existem vários tipos de sistemas implementados com etiquetas RFID, como o de acesso a edifícios via cartões que são lidos por proximidade, acesso a metrôs via cartão de pagamento, chaves de ignição de carros, pedágios, transporte público, alocação de pessoas em parques de diversão a partir de braceletes que podem ser reinicializados e reutilizados [57], etc. Mais recentemente, já em uso na Malásia, um sistema RFID está implementado em passaporte [37]. O uso em bibliotecas encontra-se também bastante difundido de modo a automatizar o controle de empréstimo, de recebimento e de inventário de livros (etiquetas para bibliotecas podem ser lidas a uma distância de 2-4 pés (13.57 MHz), Good et al. [28]). Várias bibliotecas nos Estados Unidos já adotaram a tecnologia RFID, Avoine [5]: *Santa Clara Library in California, The University of Nevada, the Las Vegas Library*, e *the Eugene Oregon Public Library.* Além da *Heiloo*, na Holanda, *Richmond Hill*, no Canadá, e *K. U. Leuven*, na Bélgica, Avoine et al. [6]. Molnar e Wagner [50] analisaram esse tipo de sistema com o objetivo de mensurar o grau de segurança das bibliotecas com sistemas RFID. Outras aplicações diferenciadas para estas etiquetas são a sua implantação em animais, para controle da origem de carnes, e em animais de criação, para melhor controle pelos seus donos quando perdidos ou em petshops. Uma outra dessas etiquetas é a ViriChip [18] que permite que prontuários médicos sejam armazenados nas referidas etiquetas. Aplicações futuras interessantes podem ser realizadas na área de triagem de material reciclável.

O Avanço da tecnologia RFID tem criado uma área ainda pouco explorada: a análise e a proposta de protocolos para dispositivos com baixo poder computacional. É claro que, no caso de produtos de maior valor, sempre existe a possibilidade do uso de soluções tradicionais de criptografia forte. Citamos, como exemplo, a solução proposta por Juels e Pappu [38] para a inclusão de RFIDs em cédulas européias, como proposto pelo Banco Central Europeu [61], [4]. O protocolo é composto pelo criptosistema El Gamal [23], codificado sob Curvas Elípticas, utilizando-se parâmetros com 40 bytes. O protocolo proposto cifra o número de série das notas e o grava no chip, que é manufaturado usualmente de silício.

2.1.2 Perspectivas Futuras de Uso e Custos

Atualmente, o custo de um sistema com RFID ainda não está suficientemente barato para substituir o Código Universal de Produto (*Universal Product Code* - UPC), conhecido como Código de Barras (óptico), criado em 1973 [19] e atualmente impresso em quase todos os produtos. Em todo o mundo, estima-se que são digitalizados mais de cinco bilhões de códigos de barra [29] diariamente. Há uma expectativa da comunidade internacional de que em breve ocorrerá uma

substituição rápida do Código de Barras pelas etiquetas, estima-se também que o custo unitário da etiqueta RFID vai baixar bastante. Assim, estima-se hoje que, em pouco tempo, esta tecnologia estará presente no dia a dia da população, estando afixadas em produtos de consumo diário. Esta tecnologia trás consigo pelo menos duas vantagens: identificação única de cada item de cada produto e também de automação. Enquanto o Código de Barras identifica apenas o tipo de objeto, uma etiqueta RFID pode emitir um número serial que distingue um objeto entre todos os outros identicamente manufaturados, além de permitir um gerenciamento de estoque mais eficiente. Em termos operacionais, o Código de Barras é ineficiente porque exige leitoras ópticas corretamente posicionadas, sendo em geral com intervenção humana, enquanto que leitoras de RFID podem realizar essas leituras a uma taxa de centenas por segundo. White [65] estima que em um futuro próximo, nos caixas de mercado poderão ser lidos automaticamente, quase que simultaneamente, aos olhos do consumidor, todos os ítens contidos em uma cesta ou carrinho.

Dispositivos RFID adequados para substituir os Códigos de Barras são conhecidos como Código Eletrônico de Produto (*Electronic Product Code* - EPC), são de custo não muito elevado, possuem um desenho de engenharia bastante simplificado, e em comunicação com a leitora emitem um identificador estático de 96 a 256 bits [16]. Infelizmente, o seu custo ainda continua proibitivo quando comparado ao custo de uma solução baseada em códigos de barra. Em 2008 o EPC tinha um custo aproximado de dez centavos de dólar por unidade, quando encomendado para fabricação em larga escala [2]. Pesquisas para obtenção de etiquetas de baixo custo é um trabalho que tem sido desenvolvido por vários centros de excelência em pesquisa. como por exemplo o MIT Auto-ID Center [49]. No entanto, os requisitos imprescindíveis para melhor aceitação e aplicação no mercado são não só o baixo custo mas também o tamanho do chip. Estas restrições implicam em impor limites de projeto, com relação ao consumo de energia, ao tempo de processamento, ao espaço de armazenamento de dados e a quantidade de portas lógicas do processador [4].

Com a expectativa desse crescente aumento no uso de etiquetas RFID, embutidas nos mais diversos produtos, espera-se que o seu preço baixe para cinco centavos em poucos anos, segundo Sarma [58], e ao mesmo tempo espera-se também que a quantidade de portas lógicas aumente para um número entre 5000 a 10000. Com a intenção de incrementar a segurança do sistema como um todo, Sarma et al. [59] foram os primeiros a expor a preocupação quanto aos riscos contra a segurança e a privacidade quando estes sistemas estiverem largamente utilizados. Por se pretender que o chip RFID tenha o menor custo possível, sua arquitetura é muito simples, deixando estas etiquetas vulneráveis a ataques como o de clonagem ou o de *skimming*. Juels [36] apresenta técnicas para torná-las mais resistentes a estes ataques, defende o uso de etiquetas EPC para combater uma grande quantidade de falsificações de produtos, mesmo embora estas etiquetas sejam vulneráveis à certos ataques criptográficos.

2.1.3 Etiqueta RFID com Criptografia

A possibilidade de implementação de criptografia robusta, padronizada, em sistemas de RFID é uma opção que deve ser considerada. Mesmo havendo perda em desempenho devido a maior quantidade de processamento necessária, a implementação de algoritmos e de protocolos criptográficos pode tornar a comunicação segura, elevando muito o grau de privacidade do usuário. Evidente-

mente, o preço das etiquetas deverá ser considerado não só pelos fabricantes e empresários, mas também pelos usuários quando compararem o custo das etiquetas RFID com e sem criptografia, ou também com àquelas que permitem apenas protocolos com criptografia mais simplificada, não padronizada. Uma análise semelhante foi feita por Abadi et al. [1] que verificou que o custo das etiquetas que comportam criptografia pode variar entre cinquenta centavos a um dólar. Estas estimativas de preços são para etiquetas com suporte à primitivas criptográficas básicas, ou com alguma resistência a violação, conforme Weis et al. [64].

2.1.4 Falhas e Ataques

Durante um processo de autenticação de uma etiqueta, na execução de um protocolo, podem ocorrer falhas. Estas falhas podem ou não colaborar com um atacante com a obtenção de alguma informação em relação ao protocolo, ao algoritmo ou a chave implementados no chip da etiqueta. Boneh et al. [13] definiram as falhas que podem ocorrer em um sistema RFID em três tipos: transientes, latentes e induzidas. Para que as falhas por indução ocorram considera-se que o adversário tem acesso físico ao dispositivo, as falhas latentes são bugs de *hardware* ou *software*, e as falhas transientes podem ser inclusive erros operacionais.

As falhas induzidas ocorrem devido a ausência de mecanismos apropriados nos chips RFID que dificultem o acesso do atacante à área de memória, Anderson e Kuhn [3]. Bar-El et al. [7] implementaram ataques por falha por indução que alteram o conteúdo da memória de um registrador, ou de uma variável específica. Alguns ataques utilizam-se de engenharia reversa como forma de recuperar o código do algoritmo, outros ataques exploram as sequências de bits trocadas entre a etiqueta e a leitora em um ou mais processos de autenticação (ataque passivo), em outros, o tempo de execução de determinada instrução do algoritmo criptográfico (*timing attack*), Kocher [45], ou a energia gasta para execução de uma determinada instrução (*power attack*), Kocher et al. [44], ou a partir da indução de falha em área de memória (*fault attack*), Boneh et al. [13], e, Kaliski e Robshaw [34]. Naturalmente, em todos esses ataques há a necessidade de se realizar análises dos resultados obtidos.

Independentemente dos protocolos criptográficos implementados nas etiquetas, existem dois outros ataques possíveis que devem ser considerados: (i) quando se utilizam dois dispositivos adicionais distintos, um deles é colocado próximo a etiqueta e o outro próximo a leitora; a comunicação entre esses dispositivos gera a aparência que tanto a etiqueta quanto a leitora estão próximos um do outro, quando na verdade podem estar bem longe (*relay attack*), Kfir e Wool [43]; ou (ii) quando uma terceira parte intermedeia a comunicação entre a etiqueta e a leitora, i.e., recebe uma informação e repassa outra, nos dois sentidos, de tal forma que ao final do protocolo possa extrair de forma direta ou lógica informação suficiente para poder personificar um dos agentes válidos, este ataque é conhecido como *man-in-the-middle attack*.

Montar um ataque real por falha pode não ser uma tarefa tão difícil, porém, é com certeza um trabalho que deve ser executado posteriormente a uma análise preliminar detalhada do alvo a ser atacado, i.e., do algoritmo e do protocolo embutidos no dispositivo. Nessa análise devem ser explicitados todos os detalhes do ataque, incluindo não só uma descrição detalhada das áreas de memória onde se pretende provocar essas falhas, mas também quais passos da análise devem ser

executados após a obtenção de resultados a partir do mal funcionamento do dispositivo. Deve-se acrescentar a esses procedimentos, com o objetivo de otimizar o ataque, o cuidado de se estabelecer um modelo de ataque que privilegie o menor número de falhas possível no dispositivo de tal forma que se obtenha o máximo de informação.

Vários ataques por falhas têm sido realizados nesses últimos anos nos mais diversos dispositivos e também em diversos algoritmos criptográficos. Em 1996, Biham e Shamir [9] propuseram um ataque por falha como método de determinação de algoritmos criptográficos implementados e encapsulados em hardware. O primeiro ataque por falha a algoritmos de bloco foi realizado no Data Encryption Standard - DES - por Biham e Shamir [10] em 1997, eles utilizaram três textos cifrados, provocando falhas em posições específicas da área de memória. Posteriormente, Giraud e Thiebeauld [26] obtiveram um resultado otimizado, utilizando-se de apenas dois textos cifrados em um ataque real em uma implementação do DES em *smart card*. O primeiro ataque por falha a algoritmo assimétrico foi proposto por Boneh et al. [13] e [14] ao algoritmo RSA-CTR, de Quisquater e Couvreur [54], uma versão variante do RSA, proposto por Rivest et al. [56]. Esta versão variante RSA-CTR é baseada no Teorema Chinês do Resto, e foi criada com objetivo de se otimizar o processo de decifração. O primeiro ataque por falha a algoritmo assimétrico baseado em curvas elípticas, especificamente em multiplicação sobre curvas elípticas, foi proposto por Biehl et al. [8]. O primeiro ataque por falha a algoritmos sequenciais ocorreu só em 2004, foi proposto por Hoch e Shamir [31] em cima de várias arquiteturas padrão de construção desses algoritmos.

2.2 Família de Protocolos HB

Um problema fundamental no uso seguro de etiquetas RFID é o da identificação de usuários. Dentre as diversas propostas apresentadas para prover segurança no processo de identificação de etiquetas, destaca-se o protocolo HB, proposto por Hopper e Blum [32] e [33], assim como os seus protocolos variantes. Estes são a base das análises realizadas neste trabalho de pesquisa de protocolos e de algoritmos criptográficos. Estes protocolos têm desempenho bastante eficientes para dispositivos de baixo custo, requerem apenas pouca área de memória para armazenamento de uma chave com k bits, compartilhada entre a leitora e a etiqueta, e utilizam operações relativamente simples, como o ou-exclusivo. A segurança do protocolo HB é baseada na dificuldade de se resolver o *learning parity with noise problem*, problema LPN, Blum et al. [11], Blum et al. [12], Hastad [30], Hopper e Blum [32] e [33], Katz e Smith [41], Kearns [42], e Regev [55].

O protocolo HB é seguro apenas para ataques passivos. Neste caso, o de adversários passivos, quebrar o protocolo HB é equivalente a resolver o problema LPN. Um dos métodos mais eficientes para a resolução do problema LPN é o proposto por Blum, Kalai e Wasserman (BKW) que tem complexidade assintótica de execução próxima a $2^{O\left(\frac{k}{\log_{10} k}\right)}$, Blum et al. [12]. Procurando minimizar os ataques correntes, e buscando um protocolo mais eficiente, Hopper e Blum [33] também propuseram o protocolo baseado na "Soma de k mínimos" com o objetivo de ser seguro também para ataques ativos.

Juels and Weis [39] modificaram o protocolo HB, propondo uma nova versão, o protocolo HB^{+}, com estrutura semelhante, agora com duas chaves compartilhadas e também com geração de mais

r desafios, estes por parte da leitora, em cada autenticação. Este protocolo HB$^+$ também foi alterado quanto ao início de cada autenticação, agora é a etiqueta que começa o processo, e não a leitora. O cálculo do produto interno é o resultado das duas chaves com os dois desafios, somados a um ruído. Gilbert et al. [24] e Katz e Shin [40] realizaram ataques ativos ao HB$^+$, mostrando que este protocolo também não é seguro contra ataques do tipo *man-in-the-middle*. Posteriormente, Carrijo et al. [17] propuseram um ataque passivo ao protocolo HB$^+$, mostrando a equivalência desses dois protocolos no ponto de vista criptoanalítico.

Seguido dos ataques de Gilbert et al. [24] e das provas de segurança apresentadas por Katz e Shin [40], Bringer et al. [15] propuseram um outro protocolo, o protocolo HB^{++}, versão alterada do protocolo HB$^+$.

Munilla e Peinado [51] apresentaram um outro protocolo, variante do HB$^+$, o protocolo HB-MP. Este protocolo tem duas chaves com k bits cada, tem desempenho mais eficiente e, segundo os autores, é resistente a ataque ativo e ao *man-in-the-middle attack*. Neste protocolo HB-MP, a leitora gera r desafios com m bits cada e a etiqueta gera outros r desafios também com m bits cada ($m \leq k$). A etiqueta desloca a primeira chave utilizando-se do i-ésimo bit da segunda chave, e calcula o produto interno entre a primeira chave deslocada e o desafio gerado pela leitora, em seguida soma-se este resultado ao ruído, obtendo-se z. Em seguida, a etiqueta escolhe um segundo desafio tal que o produto interno desse desafio com a chave deslocada seja igual a z, enviando o segundo desafio para a leitora. O protocolo tem r etapas até que possa ocorrer a autenticação.

Xuefei Leng et al. [46] propuseram uma nova versão para o protocolo HB-MP, o HB-MP$^+$. A diferença básica é que a etiqueta passa a ter uma chave que é atualizada a cada etapa do processo de autenticação. Sendo x a chave compartilhada e a_i o desafio no instante i, a chave a ser utilizada para cálculo do produto interno na i-ésima etapa é $x_i = f(a_i, x)$.

2.3 Descrição dos Protocolos da Família HB

Descrevemos aqui, de maneira detalhada, para conveniência do leitor, os protocolos da família HB que são analisados nesta tese.

2.3.1 O Protocolo HB

Considere o problema clássico de autenticação. Há duas partes, a leitora e a etiqueta, conectadas por um canal inseguro. Assumimos que a leitora e a etiqueta compartilham entre si uma informação secreta, a chave. Em um protocolo de autenticação, a leitora e a etiqueta trocam mensagens sob este canal inseguro de tal forma que, ao final desse protocolo, eles estão seguros que estão falando um com o outro. Informalmente, o protocolo é dito ser seguro se uma terceira parte maliciosa não pode personificar tanto a leitora como a etiqueta.

Este problema se torna particularmente difícil quando uma das partes tem um dispositivo com baixo poder computacional, como smart cards, RIFIDs, etc. O protocolo HB, proposto por Hopper e Blum [33], foi idealizado como uma proposta adequada para se obter autenticação segura a partir desses tipos de dispositivos.

Assume-se que a leitora e a etiqueta tenham compartilhados previamente a chave x com k bits. A descrição completa do protocolo HB é dada por:

Protocolo HB

1. Para $i = 1$ até r
   (a) A leitora gera uma sequência aleatória $\mathbf{a}_i \in \{0,1\}^k$, com k bits, e a envia para a etiqueta.
   (b) A etiqueta calcula $z_i = \mathbf{a}_i \odot \mathbf{x} \oplus \nu_i$ onde $\odot$ é o produto interno, ν_i é um bit igual a 1 com probabilidade $\eta \in (0, 1/2)$ e a soma é módulo 2. A etiqueta envia z_i para a leitora.
   (c) A leitora calcula $z_i^* = \mathbf{a}_i \odot \mathbf{x}$ e o compara com $\mathbf{z}_i$.
2. A leitora aceita a autenticação como válida se $z_i^* \neq z_i$ em menos que ηr etapas.

Neste caso, podemos considerar que r é a quantidade de equações $\{a, z\}$ definida em uma matriz de dimensão $r \times (k + 1)$.

2.3.2 O Protocolo HB$^+$

Como o objetivo de tornar o protocolo HB seguro contra ataques ativos, Juels e Weis [39] apresentaram o protocolo HB$^+$. Mostraremos, posteriormente que ambos os protocolos são semelhantes em suas estruturas com relação ao nosso ataque, e consequentemente em suas fraquezas, e podem ser quebrados de forma similar, utilizando-se do nosso método de ataque probabilístico para parâmetros adequados.

No protocolo HB$^+$ são utilizadas duas chaves x e y, secretas e compartilhadas, com k bits cada, e a etiqueta gera um bit como ruído aleatório. A descrição completa do protocolo HB$^+$ é dada por:

Protocolo HB$^+$

1. Para $i = 1$ até r
   (a) A etiqueta gera uma sequência aleatória $\mathbf{a}_i \in \{0,1\}^k$, com k bits, e a envia para a leitora.
   (b) A leitora gera uma sequência aleatória $\mathbf{b}_i \in \{0,1\}^k$, com k bits, e a envia para a etiqueta.
   (c) A etiqueta calcula $z_i = \mathbf{a}_i \odot \mathbf{x} \oplus \mathbf{b}_i \odot \mathbf{y} \oplus \nu_i$ onde $\odot$ é o produto interno, ν_i é um bit igual a 1 com probabilidade $\eta \in (0, 1/2)$ e a soma é módulo 2. A etiqueta envia z_i para a leitora.
   (d) A leitora calcula $z_i^* = \mathbf{a}_i \odot \mathbf{x} \oplus \mathbf{b}_i \odot \mathbf{y}$ e o compara com $\mathbf{z}_i$.
2. A leitora aceita a autenticação como válida se $z_i^* \neq z_i$ em menos que ηr etapas.

2.3.3 O Protocolo HB^{++}

O Protocolo HB^{++} compreende basicamente de r etapas onde a leitora e a etiqueta trocam desafios entre si, mantendo-se de forma secreta e compartilhada dois pares de chaves. Descreveremos a seguir a notação e definição mínima das variáveis utilizadas no Protocolo HB^{++}, para melhor entendimento deste protocolo:

1. A leitora e a etiqueta RFID compartilham dois pares chaves secretas $\{x, y, x', y'\}$, com k bits cada,
2. As sequências a_j, com k bits, são geradas pela leitora e enviadas para a etiqueta, j representa a j-ésima etapa de um processo de autenticação,
3. As sequências b_j, com k bits, são geradas pela etiqueta e enviadas para a leitora, j representa a j-ésima etapa de um processo de autenticação,
4. A função f é uma função hash,
5. $w \odot x$ representa o produto interno entre w e x,
6. ν e ν' são bits de ruídos gerados pela etiqueta, $\nu = 1$ ou $\nu' = 1$ tem probabilidade $\eta \in [0, 1/2)$,
7. $w \oplus \nu$ e $w \oplus \nu'$ representam o xor entre w e ν ou ν',
8. A quantidade r de etapas para a leitora autenticar a etiqueta deve variar conforme for a taxa de ruído η. Não há quantidades pré-fixadas, tanto r quanto η são pré-fixadas ao se implementar o protocolo,

O protocolo HB^{++} é aqui representado em sua j-ésima etapa de um processo de autenticação. Uma vez fixado o número de etapas r para a leitora autenticar a etiqueta, os passos devem ser executados r vezes:

Protocolo HB^{++}, j-ésima etapa de r etapas:

1. A leitora gera uma sequência aleatória a_j, com k bits, e a envia para a etiqueta,
2. A etiqueta gera uma sequência aleatória b_j, com k bits, e a envia para a leitora,
3. A etiqueta executa os seguintes comandos:
   (a) Gera os bits ν_j e ν'_j,
   (b) Calcula $z_j = a_j \odot x \oplus b_j \odot y \oplus v_j$,
   (c) Calcula $z'_j = f(a_j) \odot x' \oplus f(b_j) \odot y' \oplus v'_j$,
   (d) Envia z_j e z'_j para a leitora.
4. Ao receber z_j e z'_j da etiqueta, a leitora executa os seguintes comandos:

(a) Calcula $w_j = a_j \odot x \oplus b_j \odot y$,

(b) Calcula $w'_j = f(a_j) \odot x' \oplus f(b_j) \odot y'$,

(c) Verifica se $w_j = z_j$, soma a um contador uma unidade.

(d) Verifica se $w'_j = z'_j$, soma a um outro contador uma unidade.

5. Retorna ao item 1 por r etapas. Ao final, a leitora autentica a etiqueta caso cada contador esteja próximo a $(1 - \eta)r$.

2.3.4 Protocolos Random-HB$^{\#}$ e HB$^{\#}$

O protocolo HB$^{\#}$ difere do Random-HB$^{\#}$ apenas na forma que se monta as matrizes referentes às chaves - as chaves para esses protocolos são essas matrizes. Para o protocolo Random-HB$^{\#}$ as matrizes são montadas com sequências aleatórias, e para o protocolo HB$^{\#}$ estas matrizes são matrizes de Toeplitz. Descreveremos a seguir a notação e definição mínima das variáveis utilizadas nesses dois protocolos.

1. A leitora e a etiqueta RFID compartilham as chaves secretas X e Y, onde X é uma matriz binária com dimensão $k_x \times m$, e Y é uma matriz com dimensão $k_y \times m$,
2. A sequência a, a^{k_x}, com k_x bits, é gerada pela leitora e enviada para a etiqueta,
3. A sequência b, b^{k_y}, com k_y bits, é gerada pela etiqueta e enviada para a leitora,
4. ν, ν^m, são m bits de ruído gerados pela etiqueta, $\nu_j = 1$ tem probabilidade $\eta \in [0, 1/2)$,
5. $w \odot x$ representa o produto interno entre w e x,
6. $w \oplus \nu$ representa o xor entre w e ν,

Os protocolos Random-HB$^{\#}$ e HB$^{\#}$ são aqui representados em sua única etapa para que ocorra uma autenticação:

Protocolos Random-HB$^{\#}$ e HB$^{\#}$:

1. A leitora gera uma sequência aleatória a, e a envia para a etiqueta,
2. A etiqueta executa os seguintes comandos:

   (a) Gera uma sequência b,

   (b) Gera m bits de ruido ν,

   (c) Calcula $Z = a \odot X \oplus b \odot Y \oplus \nu$, onde Z tem dimensão m,

   (d) Envia Z e b para a leitora,

3. Ao receber Z e b da etiqueta, a leitora executa os seguintes comandos:

   (a) Calcula $Hwt(a \odot X \oplus b \odot Y \oplus Z)$, onde Hwt representa da distância de Hamming,

   (b) Verifica se $Hwt(a \odot X \oplus b \odot Y \oplus Z) \leq u \times m$, onde $u \in \,]\, \eta, \frac{1}{2}\, [$, e Hwt representa da distância de Hamming,

4. Em apenas um passo, é feita a autenticação, ou não, com m chaves distintas,

2.3.5 Protocolo HB*

O Protocolo HB* compreende basicamente de r etapas onde a leitora e a etiqueta trocam desafios entre si, mantendo-se de forma secreta e compartilhada um terno de chaves. Este protocolo é uma evolução do protocolo HB$^+$ executado em duas etapas. Descreveremos a seguir a notação e definição das variáveis utilizadas no Protocolo HB*.

1. A leitora e a etiqueta RFID compartilham as chaves secretas $\{x, y, s\}$, com k bits,
2. As sequências b_j, com k bits, são geradas pela etiqueta e enviadas para a leitora, j representa a j-ésima etapa de um processo de autenticação,
3. As sequências a_j, com k bits, são geradas pela leitora e enviadas para a etiqueta, j representa a j-ésima etapa de um processo de autenticação,
4. $w \odot x$ representa o produto interno entre w e x,
5. ν_j é um bit de ruído gerado pela etiqueta, $\nu = 1$ tem probabilidade $\eta \in [0, 1/2)$,
6. γ_j é um bit de ruído gerado pela etiqueta, $\gamma = 1$ tem probabilidade $\eta' \in [0, 1/2)$,
7. $w \oplus \nu_j$ representa o xor entre w e ν_j,
8. A quantidade r de etapas para a leitora autenticar a etiqueta deve variar conforme for a taxa de ruído η e η'. Não há quantidades pré-fixadas, tanto r quanto η e η' são pré-fixadas ao se implementar o protocolo,

O protocolo HB* é aqui representado em sua j-ésima etapa e u-ésima autenticação. Uma vez fixado o número de etapas r para a leitora autenticar a etiqueta, os passos abaixo devem ser executados r vezes:

Protocolo HB*, j-ésima etapa de r etapas:

1. A etiqueta:

   (a) Gera uma sequência aleatória b_j,

(b) Gera o bit γ_j,

(c) Calcula $w_j = b_j \odot s \oplus \gamma_j$,

(d) Envia b_j e w_j para a leitora.

2. A leitora gera uma sequência aleatória a_j e a envia para a etiqueta,

3. A etiqueta executa os seguintes comandos:

(a) Gera o bit ν_j,

(b) Se $\gamma = 0$ calcula $z_j = a_j \odot x \oplus b_j \odot y \oplus \nu_j$,

(c) Se $\gamma = 1$ calcula $z_j = b_j \odot x \oplus a_j \odot y \oplus \nu_j$,

(d) Envia z_j para a leitora.

4. A leitora ao receber z_j da etiqueta executa os seguintes comandos:

(a) Calcula $b_j \odot s$,

(b) Se $w_j = b_j \odot s$, calcula $a_j \odot x \oplus b_j \odot y$ e o compara a z_j

(c) Se $w_j \neq b_j \odot s$, calcula $b_j \odot x \oplus a_j \odot y$ e o compara a z_j

2.3.6 Protocolos Baseados no Problema da Soma de K Mínimos

Protocolos baseados no problema da Soma de K Mínimos compreende basicamente em m etapas onde a leitora e a etiqueta trocam conjuntos de desafios entre si, mantendo-se de forma secreta e compartilhada uma chave com k pares de números inteiros. Descreveremos a seguir a notação e definição das variáveis utilizadas nestes protocolos:

1. A leitora e a etiqueta RFID compartilham uma chave secreta z, onde $z = \{(x_1, y_1), ..., (x_k, y_k)\}$ é um conjunto de k pares (x_i, y_i) de inteiros, módulo n,

2. υ é um conjunto com m sequências aleatórias. Cada sequência tem n inteiros módulo 10, i.e., $\upsilon_j \in \{0, ..., 9\}^n$, $j = 1, m$, e , $k \log_{10} n \leq m < \begin{pmatrix} n \\ 2 \end{pmatrix}$,

O Protocolo baseado no problema da soma de k mínimos é aqui representado em sua j-ésima etapa. Uma vez fixado o número m de etapas para a leitora autenticar a etiqueta, os passos abaixo devem ser executados m vezes:

Protocolos Baseados no problema da Soma de K Mínimos, j-ésima etapa de m etapas:

1. A etiqueta:

(a) Gera uma sequência aleatória v_j,

(b) Calcula $u_j = f(v_j, z) = \sum_{i=1}^{k} \min\{v_j[x_i], v_j[y_i]\} \bmod 10$, para $u_j \in \{0, ..., 9\}$.

(c) Envia u_j e v_j para a leitora.

2. A leitora executa os seguintes comandos:

(a) Calcula $u'_j = f(v_j, z) = \sum_{i=1}^{k} \min\{v_j[x_i], v_j[y_i]\} \bmod 10$, para $u_j \in \{0, ..., 9\}$.

(b) Verifica se v_j é igual a v'_j, se afirmativo incrementa-se o contador,

3. repete-se este processo por m vezes.

Capítulo 3

Um Novo Ataque Passivo Probabilístico aos Protocolos HB/HB$^+$

3.1 Introdução

Protocolos para autenticação com arquitetura especialmente desenhada para dispositivos com baixo poder computacional têm sido atualmente uma área ativa de pesquisa, veja Matsumoto and Imai [47], Wang et al. [62], Naor and Pinkas [52], Hopper and Blum [33], Juels and Weis [39]. Entre os vários esquemas propostos, os protocolos HB/HB$^+$ têm recebido uma atenção especial pela sua praticidade e por ter sua segurança formalmente reduzida ao problema computacional conhecido como *the Learning Parity with Noise*, LPN, proposto por Blum, Kalai e Wasserman [12]. Neste capítulo propomos um ataque a protocolos de autenticação baseados na família HB/HB$^+$.

Podemos classificar ataques contra protocolos criptográficos de autenticação como ataques passivos ou ativos. Nos ataques passivos, o adversário armazena os desafios gerados em processos de autenticação e tenta a partir dessa informação determinar a chave secreta, esses ataques têm característica semelhante a dos ataques propostos por Blum et al. [12] e Fossorier et al. [22]. Nos ataques ativos, o adversário interfere na comunicação, altera os desafios, gera as suas próprias mensagens ou provoca erros no *chip* RFID, um exemplo é o ataque proposto por Gilbert et al. [24] onde o atacante altera o desafio. Utilizando-se de um ou de outro método de ataque, o adversário deve estabelecer métodos de análise específicos com o objetivo de extrair informação suficiente para determinar os bits da chave ou personificar um dos participantes legítimos do processo de autenticação.

Naturalmente, ataques passivos bem sucedidos têm efeito devastador já que não deixam qualquer tipo de rastro. A eficiência desses ataques é usualmente medida a partir de dois parâmetros: sua complexidade computacional e a quantidade de desafios capturados que devem ser armazenados pelo atacante. Até o momento dos nossos resultados terem sido consolidados, os mais importantes ataques passivos contra o protocolo HB são baseados no método BKW, proposto por Blum et al. [12], e a proposta de otimização desse método apresentada por Fossorier et al. [22], aqui chamado de FMICM. Esses dois métodos de ataque utilizam-se de eliminação de Gauss em amostras de sequências com η-porcento de erro, tendo como restrição o conhecimento por parte do atacante

desse erro para que se possa realizar a modelagem do ataque.

Todos os ataques passivos a esses protocolos HB/HB$^+$ já apresentados na literatura são determinísticos, i.e., produzem de forma garantida resultados válidos. Propomos aqui o primeiro ataque passivo probabilístico contra os protocolos HB/HB$^+$. O nosso ataque é baseado em um paradigma completamente diferente, precisa armazenar muito menos sequências e desafios do que os outros ataques já apresentados, não tem como restrição o conhecimento prévio do erro, enquanto mantém uma razoável complexidade computacional. Nosso ataque além de eficiente abrange os seguintes protocolos: HB, HB$^+$, HB^{++}, Random-HB$^\#$ e HB$^\#$, possuindo um escopo de aplicação bem maior do que todos os ataques previamente publicados (que somente funcionavam contra os protocolos HB e HB$^+$. Adicionalmente, apresentamos também um ataque passivo, ao protocolo HB*.

Após a submissão do nosso trabalho, Carrijo et al. [17], houve a submissão de um outro trabalho realizado de forma independente apresentado por Golebiewski et al. [27].

O restante desse capítulo está organizado da seguinte forma: na seção 3.2 é apresentado, para fins didáticos, um ataque ativo, previamente conhecido, ao protocolo HB, e na seção 3.3 é apresentado o problema LPN. Na seção 3.4 apresenta-se o método de ataque BKW e na seção 3.5 o método FMICM. Na seção 3.6 nós descrevemos o nosso ataque, e na seção 3.7 é feita uma extensão ao protocolo HB$^+$. Na seção 3.11 apresentamos a relação algébrica entre os protocolos HB, HB$^+$, HB^{++} do ponto de vista do nosso ataque, e na seção 3.12 apresentamos a relação algébrica existente entre os protocolos HB$^+$ e o HB$^\#$. Na seção 3.13 é apresentado um ataque passivo ao protocolo HB*. Na seção 3.8 apresentamos a complexidade computacional do nosso ataque ao protocolo HB, e na seção 3.9 a forma de implementação deste ataque. Na seção 3.10 nós apresentamos nossos resultados e fazemos comparações destes com os já publicados na literatura. Referente a este capítulo, no Apêndice I nós realizamos uma análise baseada em gráficos gerados em MatLab. Finalmente, nós apresentamos as conclusões na seção 3.14.

3.2 Ataque Ativo Contra o Protocolo HB

O protocolo HB é claramente inseguro no caso de adversários ativos. Por exemplo, o adversário, ao inves de enviar desafios aleatórios em cada passo, pode envia desafios idênticos. Após um certo número de respostas obtidas pelo adversário, este pode, por eliminação Gaussiana, facilmente obter a cheve secreta utilizada pela partes legais. No entanto, a segurança do protocolo HB, no caso de adversários passivos, é amplamente aceita na comunidade. No decorrer deste capítulo, mostraremos que, para parâmetros práticos, a segurança do protocolo HB mesmo contra adversários passivos é questionável.

3.3 O Problema LPN

Os protocolos HB e HB$^+$ possuem sua segurança reduzida ao classico problema LPN - *Learning Parity with Noisy* - que pode ser ser definido da seguinte forma:

1. Formulação do problema LPN

   (a) x é um parâmetro de segurança, um vetor binário k-dimensional,

   (b) $A = [a_i]_{i=1}^{r}$ é uma matriz binária de dimensão k × r, e $a_i = [a_i(j)]_{j=1}^{k}$ é um vetor coluna binário k-dimensional,

   (c) $y = [y_i]_{i=1}^{r}$ e $z = [z_i]_{i=1}^{r}$ são vetores r-dimensional,

   (d) Para cada $i = 1, ..., r$, ν_i é uma realização de uma variável aleatória binária E_i, tal que $Pr(E_i = 1) = \eta$ e $Pr(E_i = 0) = 1 - \eta$. Todas as variáveis aleatórias E_i são mutuamente independentes.

2. Dados A e x, os vetores y e z são definidos por: $y = x \odot A$ e $z_i = y_i \oplus \nu_i$, para $i = 1, 2, ..., r$.

3. Dados (A, y), x é facilmente determinado, basta utilizar-se de técnicas de álgebra linear. Dados (A, z, η), recuperar x é um problema mais difícil, este é o problema LPN. O problema LPN se torna mais complexo quanto maior for η.

Uma prova de segurança estabelecendo uam relação entre a quebra de segurança de um protocolo e um problema NP completo (como é o caso do LPN), é considerada uma garantia forte de que o protocolo em questão é efetivamente seguro. No entanto, devemos lembrar que, no mundo real, devemos escolher parâmetros de segurança para nossos protocolos e que a intratabilidade de certos problemas computacionais é usualmente assintótica. Logo, uma prova de segurança reduzindo a quebra de um protocolo a resolução de um problema intratável não substitui uma análise concreta do protocolo com parâmetros práticos.

3.4 O Método BKW de Ataque Passivo ao Protocolo HB

O método mais famoso para resolver o problema LPN é o algoritmo BKW [12] que tem como complexidade aproximada:

$$2^{O(\frac{k}{\log(k)})} \tag{3.1}$$

O algoritmo BKW consiste no método de se minimizar por meio de eliminação de Gauss, em um grande conjunto de pares desafios-resposta, o máximo de combinações lineares existentes. Este algoritmo pode deduzir a chave secreta com alta probabilidade a partir de repetidos ensaios e o conhecimento prévio do percentual η de ruído, para anterior modelagem do ataque.

As equações a seguir, definidas por Weis et al. [63], permitem estimar os melhores parâmetros a serem utilizados na modelagem do método BKW com o objetivo de se determinar com alta probabilidade a solução do sistema de equações não-lineares, montadas a partir de pares desafios-resposta gerados em processos de autenticação de etiquetas. O esforço computacional e a quantidade de desafios convergem para valores de mesma ordem de grandeza e podem ser estimados por:

$$\alpha \cdot \beta \geq \mathrm{k} \tag{3.2}$$

$$M = \max\{\left(\frac{1}{1-2\cdot\eta}\right)^{2.\beta}, \beta\} \tag{3.3}$$

$$t = C \cdot \alpha^3 \cdot M \cdot 2^{\beta} \tag{3.4}$$

α e β devem ser escolhidos de modo a minimizar t. Por exemplo, utilizando-se dessas equações, para k=32 e $\eta = 0.25$, os valores que minimizam t são α=2 e β=16, os quais resultam em t=2^{24}. Isto significa que em um ataque passivo são necessários aproximadamente 2^{24} desafios e processamento de 2^{24} resoluções de sistemas para se determinar a chave com 32 bits. As tabelas 3.2, 3.3, 3.4, e 3.5, seção 3.10, referentes ao método BKW, foram definidas por estes parâmetros, e mostram a relação entre o comprimento da chave, o esforço computacional e a quantidade de pares desafios-resposta necessários para se quebrar a chave do protocolo HB.

3.5 Método FMICM - Parâmetros de Otimização

Fossorier et al. [22] também propuseram um algoritmo (MFICM) para atacar o protocolo HB. Este algoritmo é resultado de um outro trabalho proposto por Fossorier et al. [21] sobre ataques por correlação a algoritmos criptográficos sequenciais, e emprega conceito sobre decimação, combinação linear, teste de hipótese, e decodificação de distância mínima. Essa proposta otimiza o esforço computacional exigido pelo método BKW, mantendo porém a quantidade de amostras que cresce exponencialmente em função de k e η.

As estimativas referentes à complexidade computacional e quantidade de amostras necessárias para se aplicar o método FMICM contidas nas tabelas 3.2, 3.3, 3.4, e 3.5, seção 3.10, são resultados da implementação das seguintes expressões [22]:

$$t \approx \begin{pmatrix} n^*/2 \\ \lceil w/2 \rceil \end{pmatrix} + 2^{b_H} \log_2(1-2\eta)^{-2w} \tag{3.5}$$

$$s \approx n^* + \begin{pmatrix} n^*/2 \\ \lfloor w/2 \rfloor \end{pmatrix} \tag{3.6}$$

Onde,

$$b_{H,otimo} = 2w \log_2(1-2\eta)^{-1} + k - b_0 - \log_2 \begin{pmatrix} n^* \\ w \end{pmatrix} \tag{3.7}$$

$$b_{H,otimo} > w \log_2(1-2\eta)^{-2} \tag{3.8}$$

$$\log_2 C_{III,otimo} \approx 2w \log_2(1-2\eta)^{-1} + k - b_0 - \log_2 \begin{pmatrix} n^* \\ w \end{pmatrix} + t_A \tag{3.9}$$

$$t_A = \log_2(k-b_0) + \log_2 w + \log_2 \log_2(1-2\eta)^{-2} \tag{3.10}$$

$$C_{II} \approx \log_2(k-b_0) + \log_2 \begin{pmatrix} n^*/2 \\ \lceil w/2 \rceil \end{pmatrix} \tag{3.11}$$

$$b_{0,otimo} \approx \left(\frac{3w}{2} - 1\right)^{-1} \left(\frac{3w}{2}\left(\log_2 n^* - \log_2 w\right) + t_B\right) \tag{3.12}$$

$$t_B = -2w \log_2(1-2\eta)^{-1} + k + \frac{w}{2} - \log_2 w - \log_2 \log(1-2\eta)^{-2} \tag{3.13}$$

$n^* = 2^{b_0} \cdot n$, e n é a quantidade de amostras utilizas pelo método BKW, veja tabelas 3.2, 3.3, 3.4, e 3.5, seção 3.10.

O método FMICM tem complexidade computacional dada por $C \approx C_{II} + C_{III}$. As expressões que definem C são inter-relacionadas, as estimativas são obtidas a partir da indicação prévia de alguns valores para b_0 e w. Na prática, a estimativa inicial de b_0 deve ser utilizada apenas como referência pontual do real b_0 ótimo a ser utilizado para se determinar os outros parâmetros que otimizam o método. O atacante ao calcular estimativas para t e s deve fazer algo como um *trade-off* entre memória e esforço computacional. Existe um conjunto de resultados otimizados, com complexidade aproximada.

3.6 Proposta de Ataque Passivo Probabilístico ao Protocolo HB

Descrevemos nessa seção nosso ataque passivo probabilístico contra o protocolo HB.

Considerando a descrição do protocolo HB, seção 2.3.1, o nosso método de ataque passivo e probabilístico assume que o adversário capturou m desafios a partir de processos de autenticação entre etiquetas e a leitora, com $m > k$ para o protocolo HB, e $m > 2k$ para o protocolo HB$^+$.

Observa-se que, tanto no protocolo HB quanto em sua extensão HB$^+$, caso não ocorra a introdução de ruído por uma das partes, a chave pode ser obtida por uma simples eliminação Gaussiana. Nota-se também que a introdução do ruído é aleatória. Portanto, dentro de um conjunto de m pares de desafios-resposta, haverá um subconjunto de pares não contaminados por ruído. Obviamente, um adversário não consegue diferenciar entre um par contaminado por ruído e outro par livre de ruído. Nossa estratégia é escolher um subconjunto de pares desafio-resposta aleatoriamente e montar um conjunto de equações lineares para resolve-lo. Caso uma solução exista, pode-se testá-la com a ajuda dos pares não utilizados na montagem das equações.

De maneira mais detalhada: Seja A uma matriz de dimensão $m \times k$; $[a_i]_{i=1}^{m}$ é um vetor da i-ésima linha de A. Cada linha dessa matriz representa um par desafio-resposta de um processos de

autenticação utilizando este protocolo. Seja ν um vetor coluna m-dimensional onde cada entrada representa um bit de ruído, ν_i. Dado x, um vetor k-dimensional, define-se por z como $z = A \odot x \oplus \nu$, um vetor m-dimensional. Também podemos assumir que a distância de Hamming de ν não será maior do que 0.40m (caso contrário o procedimento de autenticação pode se tornar frequentemente falível), e m>k. Seja C um subconjunto de $\{1, ..., m\}$ desafios com cardinalidade $|C|$. Seja $C(i)$ o i-ésimo elemento de C e a matriz $\left[a_{C(i)}\right]_{i=1}^{|C|}$ representada por A_C. Seja $x(i)$ o i-ésimo elemento do vetor coluna x, e o vetor coluna $\{x(C(1)), ..., x(C(|C|))\}$ é representado por x_C.

Entrada do Algoritmo (ataque) Passivo Probabilístico ao HB: (A, z)

1. Selecione aleatoriamente um subconjunto C com cardinalidade $n = k + \gamma$ (γ é um inteiro sutilmente escolhido),
2. Calcule, utilizando eliminação de Gauss, x_C tal que $z_C = A_C \odot x_C$. Se não existir solução, ou se houver múltiplas soluções, retorne ao passo anterior,
3. Cheque a distância de Hamming, se $A \odot x_C$ for maior do que $0.40m$ retorne ao primeiro passo, caso contrário x_C é a solução procurada.

3.7 Extensão da Proposta para Ataque ao Protocolo HB$^+$

O nosso ataque pode ser trivialmente estendido ao protocolo HB$^+$ utilizando-se de uma quantidade de desafios duas vezes maior, uma vez que este protocolo utiliza-se de duas chaves. Seja $x \,||\, y$ a concatenação de dois vetores coluna k-dimensional x e y tal que $x||y = (x_1, ..., x_k, y_1, ..., y_k)$. Seja $A \,||\, B$ a concatenação de duas matrizes A e B de dimensão $m \times k$ tal que $A||B = [a_i||b_i]_{i=1}^{m}$. Seja z igual a z = [A||B][x||y] $\oplus \nu$, e ν é definido como anteriormente.

Entrada do Algoritmo (ataque) Passivo Probabilístico ao HB$^+$: (A, B, z):

1. Selecione aleatoriamente um subconjunto C com cardinalidade $n = 2k + \gamma$ (γ é um inteiro sutilmente escolhido),
2. Calcule, utilizando eliminação de Gauss, $[x||y]_C$ tal que $\mathrm{z_C}$= $[\mathrm{A||B}]_\mathrm{C} \odot [\mathrm{x||y}]_\mathrm{C}$. Se não existir solução, ou se houver múltiplas soluções, retorne ao passo anterior,
3. Cheque a distância de Hamming, se [A||B]$\odot[x||y]_C$ for maior do que $0.40n$ retorne ao primeiro passo, caso contrário $[x||y]_C$ é a solução procurada.

3.8 Complexidade Computacional do Nosso Ataque

A complexidade do nosso ataque probabilístico é facilmente definida, uma vez que procuramos minimizar a quantidade de amostras que formam o nosso universo, realizando experimentos sem

reposição, procurando obter alta probabilidade de sucesso no nosso ataque, i.e., determinando-se a chave. Basicamente, o nosso ataque passivo consiste em: o adversário captura e armazena m pares de desafios-resposta, correspondentes a equação linear da forma $z_i = a_i \odot x_C$, e cria sistemas lineares $z_C = A_C \odot x_C$ com n desses pares $\{a_i, z_i\}$ escolhidos de forma aleatória.

Existem m equações a serem escolhidas, onde em média $(1-\eta)m$ equações são corretas e ηm incorretas. A probabilidade p de se quebrar o criptosistema é a probabilidade de se escolher n equações corretas (livres de ruído) e linearmente independentes, entre as m disponíveis, em experimentos sem repetição. Essa probabilidade pode ser expressa por:

$$p = \frac{\begin{pmatrix} (1-\eta)\cdot m \\ n \end{pmatrix}\begin{pmatrix} \eta \cdot m \\ o \end{pmatrix}}{\begin{pmatrix} m \\ n \end{pmatrix}} = \frac{\begin{pmatrix} (1-\eta)\cdot m \\ n \end{pmatrix}}{\begin{pmatrix} m \\ n \end{pmatrix}} \tag{3.14}$$

Portanto, em média, é necessário resolver $1/p$ resoluções de sistemas lineares para recuperar a chave secreta e compartilhada x. Consequentemente, a complexidade média esperada desse ataque é dada por:

$$\frac{1}{p} = \frac{\begin{pmatrix} m \\ n \end{pmatrix}}{\begin{pmatrix} (1-\eta)\cdot m \\ n \end{pmatrix}} = \frac{m \cdot (m-1) \cdot ... \cdot (m-n+1)}{\eta \cdot m \cdot (\eta \cdot m - 1) \cdot ... \cdot (\eta \cdot m - n + 1)} \tag{3.15}$$

Como ilustrado na equação 3.15, o esforço computacional depende basicamente dos parâmetros $\{\eta, m, n\}$, onde (para o caso do HB) $n = k + \gamma$, e $\gamma \geq 0$. Esse esforço computacional aumenta quando: (i) η aumenta, ou; (ii) quando o número de pares desafios-resposta m é muito próximo ao comprimento k da chave secreta x, ou; (iii) se γ não for próximo a zero quando $k < 100$, ou; (iv) se γ for superior a 10 ou 15% do valor de k, para $k > 100$. É importante entender que por ser um ataque probabilístico, sob condições similares e com os mesmos parâmetros, diferentes execuções desse algoritmo criptoanalítico e probabilístico pode resultar em valores indicativos de complexidade pontuais e diferentes entre si, caracterizando a natureza probabilística. Nesta equação 3.15 também não se considera o fator multiplicador que é o inverso da probabilidade de se encontrar pelo menos uma equação linearmente dependente entre n equações escolhidas, por ser irrelevante no nosso ataque.

Foi implementado em linguagem C um módulo que determina o esforço computacional necessário para se quebrar o protocolo HB, e de forma equivalente o protocolo HB$^+$. Este módulo estima em média quantas resoluções de sistemas lineares são necessários para se recuperar a chave correta com k bits, dado η e m. Estas estimativas estão contidas nas tabelas 3.2, 3.3, 3.4, e 3.5, seção 3.10.

3.9 Implementação do Método Proposto de Quebra dos Protocolos HB e HB$^+$

O esforço computacional e a quantidade de amostras contidos na Tabela 3.1, seção 3.10, são resultados da implementação em linguagem C, ambiente Windows, de um módulo que quebra os protocolos HB e HB$^+$. Este módulo que determina todos os bits da chave secreta possui duas partes: na primeira parte simula-se a geração dos desafios, fixa-se os parâmetros $\{x, \eta\}$, e assume-se que o atacante tem conhecimento exclusivo dos pares desafios-resposta. Na segunda parte, o módulo quebra o protocolo, i.e., determina a chave x, sem o conhecimento desses parâmetros $\{x, \eta\}$. A implementação desse módulo não foi otimizada, embora pudesse ter sido feita com o objetivo de se obter melhor desempenho ou uso de menos memória. Esta otimização não foi realizada porque tanto o desempenho desse módulo quanto a memória necessária para o procedimento de quebra não são fatores críticos para obtenção dos nossos resultados contidos na Tabela 3.1, seção 3.10. Dessa forma, este módulo foi implementado com a restrição de se ter no máximo 4000 desafios, e chave com no máximo 100 bits, e $\eta \in (0, 1/2)$. Essas fronteiras foram estabelecidas apenas por conveniência de implementação, uma vez que o nosso método pode ser aplicado de forma geral. Esse módulo foi desenvolvido com a seguinte estrutura de rotinas:

Implementação do nosso Método - Estrutura de rotinas:

1. Geram-se a chave secreta x e m sequências pseudo-aleatórias a_i, com k bits,
2. Geram-se ruídos ν_i, de forma pseudo-aleatória, com distribuição de Bernoulli, e probabilidade η, e somam-se estes aos resultados dos produtos internos entre a chave e cada sequência a_i, gerando-se desafios z_i,
3. Escolhem-se de forma aleatória subconjuntos com n equações ($n = k + \gamma$), entre todas as m calculadas no item 2, e retorna o par $\{a', z'\} \subset \{a, z\}$,
4. Resolvem-se sistemas de equações $\{a', z'\}$ e verifica-se a existência de solução única. Se não existir, retornar ao item 3,
5. Se ω for uma solução, verifica-se se ω é a solução correta, utilizando de todos os m pares $\{a, z\}$. Caso não seja a solução procurada, retornar ao item 3.

3.10 Análise dos Dados e Comparações

Como o nosso método de ataque é probabilístico, ao se executar o módulo de ataque pode ser observado que em algumas execuções a solução é determinada de forma bastante rápida, e em outras não, utilizando-se dos mesmos parâmetros, diferindo apenas das sementes do gerador pseudo-aleatório. O nosso método consiste em realizar sorteios aleatórios de subconjuntos de equações entre as m equações armazenadas, e em seguida tentar resolver cada um desses sistemas. Na tabela 3.1 são apresentados alguns resultados obtidos a partir de nossa impelementação:

Tabela 3.1: Cálculo prático da Complexidade Computacional, k=100 bits, n=110 equações, e $\eta = 10\%$

m : Total de desafios	Total de soluções erradas até obtenção da solução correta	Total de resoluções de sistemas até obtenção da solução correta	Tempo de obtenção, Pentium 4 1.4 GHz (em segundos)
4000	26	24.851	58
3000	63	49.070	116
2000	24	29.614	70
700	70	71.827	170
400	17	18.625	44
350	63	75.748	170
300	188	192.894	449
250	2646	2.683.810	6.237

Observando-se a Tabela 3.1, a chave secreta foi determinada com $m = 2000$ mais rapidamente do que para $m = 3000$; e com $m = 400$ mais rápido que para $m = 700$. Foi realizado também experiência semelhante para $k = 32$ e $k = 64$. Para $k = 32$, $n = 36 = 32 + \gamma$, $\eta = 0.1$, e, m variando entre 4000 e 50, determinou-se a chave, em todos os casos com menos um segundo. O mesmo ocorreu para $k = 64$, $n = 70 = 64 + \gamma$, $\eta = 0.1$, e, m variando ente 4000 e 200. A partir de execuções exaustivas dos nossos módulos de ataque, foi observado que se m for muito próximo a k, o erro calculado de uma solução errada pode convergir para um valor próximo a η, porém, a solução correta mantém o erro η.

Comparando-se os três métodos de ataque (o BKW, o FMICM, e o nosso) para $k = 96$ e $\eta = 0.20$, a complexidade computacional do método BKW é de 2^{42} passos, a do método otimizado FMICM é de 2^{25}, e a do nosso método é de 2^{25} passos, para obtenção de cada bit de informação da solução correta. Porém, a quantidade de pares desafios-resposta necessária para cada um desses métodos é de 2^{42}, 2^{24}, 2^{12}, respectivamente.

A quantidade de amostras necessárias para se aplicar os métodos BKW e FMICM cresce exponencialmente, sendo proporcional a $\alpha^3 \cdot M \cdot 2^{\beta}$. Aplicando-se esses métodos para se determinar a chave x, para $k = 100$ e $\eta = 0.1$, são necessários aproximadamente 2^{36} e 2^{24} pares de desafios-resposta, respectivamente, e de 2^{36} e 2^{26} passos, respectivamente, por bit de informação para se determinar cada bit da chave. Comparando a eficiência desses métodos com o nosso aqui proposto, com valores reais, recuperou-se a chave, no melhor caso, vide Tabela 3.1, com aproximadamente $2^{14}/100$ passos, por bit de informação, 44 segundos de processamento, utilizando-se de apenas 400 pares de desafios-resposta.

Nós comparamos a complexidade computacional e a quantidade de pares de desafio-resposta necessários para se quebrar o protocolo HB, e de forma equivalente o HB$^+$. Esta comparação foi feita entre o nosso método, o BKW e o FMICM. É importante enfatizar que tanto o BKW

quanto o FMICM são ataques determinísticos, enquanto o nosso é probabilístico. Dessa forma, as complexidades aqui apresentadas do nosso ataque são valores esperados, assim como a complexidade computacional para esses ataques são apresentadas por bit de informação, como apresentado também em Juels e Weis [39] e em Fossorier et al. [22]. Para detalhes de desempenho dos métodos BKW e FMICM nós nos referenciamos a [12] e [22], respectivamente.

Foram feitas várias simulações gerando várias tabelas e gráficos variando m, n e η. Os objetivos foram não só o de se comparar os resultados tabelados referentes aos três métodos de ataque, mas também o de verificar a convergência das curvas que poderiam definir uma quantidade otimizada de desafios e esforço computacional necessários para se quebrar os protocolos HB e HB$^+$. Para cada parâmetro $n = k = \{32, 64, 96, 128, 160, 192, 224, 256, 288\}$, foram geradas tabelas e gráficos para $\eta = \{10\%, 15\%, 20\%, 25\%\}$. As tabelas 3.2, 3.3, 3.4, e 3.5 mostram os resultados referentes a esses parâmetros, relativos ao nosso métodos e aos métodos BKW e FMICM.

	Complexidade Computacinal			**Quantidade de pares desafios-resposta**		
Chave	**BKW**	**FMICM**	**Nosso Método**	**BKW**	**FMICM**	**Nosso Método**
32	2^{23}	2^{8}	2^{9}	2^{23}	2^{8}	2^{10}
64	2^{35}	2^{16}	2^{21}	2^{35}	2^{16}	2^{12}
96	2^{45}	2^{25}	2^{34}	2^{45}	2^{24}	2^{13}
128	2^{54}	2^{34}	2^{47}	2^{54}	2^{34}	2^{13}
160	2^{62}	2^{43}	2^{60}	2^{62}	2^{42}	2^{14}
192	2^{70}	2^{52}	2^{73}	2^{70}	2^{51}	2^{14}
224	2^{78}	2^{60}	2^{87}	2^{78}	2^{60}	2^{14}
256	2^{86}	2^{69}	2^{99}	2^{86}	2^{69}	2^{14}
288	2^{94}	2^{81}	2^{113}	2^{94}	2^{81}	2^{14}

Tabela 3.2: Comparação do esforço computacional esperado e a quantidade de pares desafios-resposta para quebra do protocolo HB; necessários para o nosso método e os métodos de ataque BKW e FMICM, em função de $\eta = 0.25$ e do comprimento da Chave.

	Complexidade Computacinal			Quantidade de pares desafios-resposta		
Chave	**BKW**	**FMICM**	**Nosso Método**	**BKW**	**FMICM**	**Nosso Método**
32	2^{22}	2^{8}	2^{6}	2^{22}	2^{9}	2^{10}
64	2^{33}	2^{16}	2^{15}	2^{33}	2^{10}	2^{12}
96	2^{42}	2^{25}	2^{25}	2^{42}	2^{24}	2^{12}
128	2^{50}	2^{35}	2^{34}	2^{50}	2^{19}	2^{13}
160	2^{58}	2^{44}	2^{45}	2^{58}	2^{42}	2^{13}
192	2^{66}	2^{52}	2^{58}	2^{66}	2^{50}	2^{13}
224	2^{74}	2^{62}	2^{65}	2^{74}	2^{60}	2^{15}
256	2^{82}	2^{71}	2^{75}	2^{82}	2^{65}	2^{15}
288	2^{89}	2^{83}	2^{86}	2^{89}	2^{83}	2^{15}

Tabela 3.3: Comparação do esforço computacional esperado e a quantidade de pares desafios-resposta para quebra do protocolo HB; necessários para o nosso método e os métodos de ataque BKW e FMICM, em função de $\eta = 0.20$ e do comprimento da Chave.

	Complexidade Computacinal			Quantidade de pares desafios-resposta		
Chave	**BKW**	**FMICM**	**Nosso Método**	**BKW**	**FMICM**	**Nosso Método**
32	2^{21}	2^{8}	2^{3}	2^{21}	2^{8}	2^{10}
64	2^{31}	2^{19}	2^{9}	2^{31}	2^{19}	2^{13}
96	2^{39}	2^{26}	2^{17}	2^{39}	2^{24}	2^{13}
128	2^{47}	2^{35}	2^{23}	2^{47}	2^{33}	2^{13}
160	2^{55}	2^{43}	2^{31}	2^{55}	2^{43}	2^{13}
192	2^{63}	2^{52}	2^{39}	2^{63}	2^{51}	2^{13}
224	2^{69}	2^{62}	2^{47}	2^{69}	2^{62}	2^{14}
256	2^{76}	2^{71}	2^{56}	2^{76}	2^{71}	2^{14}
288	2^{82}	2^{83}	2^{61}	2^{82}	2^{83}	2^{14}

Tabela 3.4: Comparação do esforço computacional esperado e a quantidade de pares desafios-resposta para quebra do protocolo HB; necessários para o nosso método e os métodos de ataque BKW e FMICM, em função de $\eta = 0.15$ e do comprimento da Chave.

	Complexidade Computacinal			Quantidade de pares desafios-resposta		
Chave	**BKW**	**FMICM**	**Nosso Método**	**BKW**	**FMICM**	**Nosso Método**
32	2^{20}	2^{8}	2^{1}	2^{20}	2^{8}	2^{10}
64	2^{28}	2^{17}	2^{4}	2^{28}	2^{16}	2^{10}
96	2^{36}	2^{26}	2^{9}	2^{36}	2^{24}	2^{11}
128	2^{44}	2^{35}	2^{13}	2^{44}	2^{33}	2^{13}
160	2^{50}	2^{44}	2^{18}	2^{50}	2^{43}	2^{13}
192	2^{57}	2^{54}	2^{24}	2^{57}	2^{51}	2^{13}
224	2^{63}	2^{62}	2^{27}	2^{63}	2^{62}	2^{13}
256	2^{70}	2^{71}	2^{31}	2^{70}	2^{71}	2^{14}
288	2^{76}	2^{85}	2^{36}	2^{76}	2^{85}	2^{14}

Tabela 3.5: Comparação do esforço computacional esperado e a quantidade de pares desafios-resposta para quebra do protocolo HB; necessários para o nosso método e os métodos de ataque BKW e FMICM, em função de $\eta = 0.10$ e do comprimento da Chave.

É importante enfatizar que métodos de ataques contra protocolos de comunicação do tipo desafio-resposta, implementados em *smart cards*, RFIDs e outros dispositivos com banda de comunicação reduzida, onde a coleta dessa informação é lenta, é muito mais eficiente e prático àqueles ataques que exijam menor quantidade de pares desafios-resposta e tenham um pouco maior a complexidade computacional. O tipo de comunicação utilizado é um limitante significativo para qualquer ataque prático a esses protocolos devido a pequena capacidade de transmissão/captura de desafios. Tais dispositivos se comunicam usualmente em taxas não tão altas (tipicamente não maiores do que 1 Mbps). Então, sabendo que em um passo do protocolo HB são transmitidos k bits, onde k usualmente tem 256 bits, e que para um ataque que necessite de 2^{80} passos do protocolo HB, um adversário terá que esperar $2^{88}/10^{6}$ segundos, que é maior do que a idade estimada do universo. Depois de capturar toda essa informação, só então o atacante poderá iniciar os cálculos para determinar a chave. Como mostrado nas tabelas 3.2, 3.3, 3.4, 3.5, o nosso ataque reduz drasticamente a quantidade de desafios necessários para se obter a chave, tornando o nosso ataque bastante prático e eficiente.

Para $\eta = \{0.20, 0.15, 0.10\}$ o nosso ataque apresenta melhor complexidade computacional e quantidade de desafios do que o BKW. Para $\eta = 0.25$ e chave com comprimento maior do que 160 bits, o BKW apresenta melhor complexidade computacional.

Para $\eta = 0.15$ e $\eta = 0.10$ o nosso método obtém menor complexidade computacional e requer menos quantidade de pares de desafios-resposta do que o FMICM. Para $\eta = 0.20$ o nosso método ainda requer muito menos desafios do que FMICM. No entanto, as complexidades computacionais dos dois métodos são semelhantes (com uma ligeira vantagem para o FMICM). Para $\eta = 0.25$ o nosso método é menos eficiente em termos da complexidade computacional quando comparada ao FMICM, mas ainda requer muito menos pares de desafios-resposta capturados a partir dos processos de comunicação.

É importante comentar que para percentuais de ruído $\eta = 0.15$ e $\eta = 0.10$ o desempenho do FMICM significativamente degrada quando comparado ao BKW e ao nosso método.

Nas seções seguintes é possível ao observar os gráficos verificar que o atacante pode determinar a chave secreta, ao aplicar o nosso método, bastando escolher as condições que lhe favoreçam, realizando um *trade-off* entre a quantidade de desafios e o esforço computacional disponíveis necessários para obtenção da solução correta.

3.11 Relação Algébrica entre os Protocolos HB $\approx$ HB^{+} $\approx$ HB^{++}

Gilbert at al. [24] propuseram um ataque do tipo *main-in-the-midlle* ao protocolo HB^{+}, em seguida Bringer et al. [15] propuseram uma nova versão a este protocolo, chamado de HB^{++}, com o objetivo de frustrar todas os ataques propostos por Gilbert et al. [24].

Em uma análise direta, em relação ao nosso ataque probabilístico, o esforço computacional necessário para se determinar as chaves secretas e compartilhadas, $\{x, y, x', y'\}$, deste protocolo HB^{++}, é de apenas duas vezes o esforço computacional necessário para se quebrar o protocolo HB^{+}, uma vez que é necessário apenas montar dois sistemas de equações independentes para se determinar z e z'.

Para se determinar z, basta utilizar-se do método de ataque proposto na seção 3.7, na sua íntegra. Porém, para determinar z', primeiro deve-se fazer a seguinte relação de equivalência: Seja $x' \,||\, y'$ a concatenação de dois vetores coluna k-dimensional x' e y' tal que $x'||y' = (x_1^{'}, ..., x_k^{'}, y_1^{'}, ..., y_k^{'})$. Seja $A' \,||\, B'$ a concatenação de duas matrizes A' e B' de dimensão $m \times k$ tal que $A'||B' = [f(a_i)||f(b_i)]_{i=1}^{m}$. Seja z' igual a $\mathrm{z}' = [\mathrm{A}'||\mathrm{B}'][\mathrm{x}'||\mathrm{y}'] \oplus \nu^{'}$, e $\nu^{'}$ é definido da mesma forma que ν. Em seguida, deve-se utilizar-se do mesmo método de ataque proposto na seção 3.7.

Na segunda versão do protocolo HB^{++}, proposta por Bringer et al. [15], foram realizadas duas modificações básicas: a primeira, as chaves $\{x, y, x', y'\}$ são geradas a partir de uma função hash que tem como entrada uma única chave compartilhada entre a etiqueta e a leitora e de um par de desafios que são trocados logo no início da comunicação. A segunda alteração mantém a mesma relação de equivalência que obtivemos ao analisar a primeira versão do HB^{++}, alterando-se exclusivamente o seguinte ponto: $A'||B' = [f(a_i, \rho)||f(b_i, \rho)]_{i=1}^{m}$, uma vez que ρ é fixo. Desa forma, seja z' igual a $\mathrm{z}' = [\mathrm{A}'||\mathrm{B}'][\mathrm{x}'||\mathrm{y}'] \oplus \nu^{'}$, e $\nu^{'}$ é definido da mesma forma que ν. Em seguida, deve-se utilizar-se do método de ataque proposto na seção 3.7.

Piramuthu [53] propôs alteração do protocolo HB^{++}, criando uma terceira versão, nesta, utilizando-se apenas um par de chave x e y. Em relação ao nosso ataque (probabilístico), o esforço computacional exigido para se determinar as chaves secretas e compartilhadas, x e y, é equivalente ao esforço computacional necessário para se quebrar o protocolo HB^{+}, uma vez que: $A'||B' = [f(a_i, \rho)||f(b_i, \rho)]_{i=1}^{m}$, e $\mathrm{z}' = [\mathrm{A}'||\mathrm{B}'][\mathrm{x}||\mathrm{y}] \oplus \nu$. Em seguida, deve-se utilizar-se do método proposto na seção 3.7.

3.12 Relação Algébrica entre os Protocolos HB$^+$ $\approx$ HB$^\#$

Gilbert el al. [25] apresentaram como propostas duas novas versões de protocolo: Random-HB$^\#$ e HB$^\#$, como evolução do protocolo HB$^+$. O protocolo HB$^\#$ difere do Random-HB$^\#$ apenas na forma que se monta as matrizes referentes às chaves - as chaves para esses protocolos são essas matrizes. Para o protocolo Random-HB$^\#$ as matrizes são montadas com sequências aleatórias, e para o protocolo HB$^\#$ estas matrizes são matrizes de Toeplitz. Do ponto de vista de nossa análise já apresentada, desconsideraremos inicialmente essa pseudo-aleatoriedade, tratando apenas como chaves aleatórias. Nós analisaremos esses dois protocolos e também procuramos estabelecer uma relação de equivalência entre estes protocolos do ponto de vista dos nossos ataques. Abaixo, oportunamente, faremos algum comentário sobre o fato de existirem chaves semelhantes, deslocadas em uma posição, em cada coluna da matriz, referente ao protocolo HB$^\#$. Para os comentários logo a seguir, utilizamos da descrição do protocolo HB$^+$, seção 2.3.2, e faremos uma breve descrição do protocolo Random-HB$^\#$ $\approx$ HB$^\#$:

Protocolos Random-HB$^\#$ e HB$^\#$:

1. $Z = a \odot X \oplus b \odot Y \oplus \nu$,
2. $\{Z^m, a^{k_x}, b^{k_y}, X^{k_x \times m}, Y^{k_y \times m}, \nu^m\}$,
3. Em apenas um passo, é feita a autenticação, ou não, com m chaves distintas, observada a restrição $Hwt(a \odot X \oplus b \odot Y \oplus \nu \oplus Z) \leq u \times m$, onde $u \in \,]\, \eta, \frac{1}{2} \,[$, e Hwt representa a distância de Hamming,
4. Considerando:
   (a) z^s é o s-ésimo elemento do vetor binário Z,
   (b) x^s é a s-ésima coluna da matriz binária X,
   (c) y^s é a s-ésima coluna da matriz binária Y,
   (d) ν^s é o s-ésimo bit do vetor binário ν, para $s = 1, m$,
   (e) e, $z^s = a^s \odot x^s \oplus b^s \odot y^s \oplus \nu^s$ corresponde a um passo do protocolo HB$^+$.
5. Os protocolos Random-HB$^\#$ e HB$^\#$ são equivalentes ao protocolo HB$^+$, baseado na seguinte relação $z^s = a^s \odot x \oplus b^s \odot y^s \oplus \nu^s = (a^s||b^s) \odot (x^s||y^s) \oplus \nu^s = d \odot q \oplus \nu$, para $d^s = a^s||b^s$ e $q^s = x^s||y^s$, onde d^s e q^s são vetores com $k_x + k_y$ bits. Portanto, para o sistema completo a relação também é semelhante: $Z = a \odot X \oplus b \odot Y \oplus \nu = (a||b) \odot (X||Y) \oplus \nu = C^{1 \times ((k_x + k_y)} \odot D^{(k_x + k_y) \times m} \oplus \nu^{1 \times m}$.

Conclui-se que os protocolos Random-HB$^\#$ e HB$^\#$ podem ser considerados equivalentes aos protocolos HB$^+$, e HB, do ponto de vista dos nossos ataques, em relação a suas variáveis, tanto o probabilístico como o por falha, este apresentado no capítulo 4. A diferença principal é que para

esses dois protocolos o atacante deverá realizar m ataques em separado, definindo as m chaves distintas de comprimento $k_x + k_y$. Além disso, ao final, o atacante deve realizar também o teste de adequação dos resultados, em função da distância de Hamming, conforme a definição desses protocolos [25], ou no item 3 de sua descrição.

Para cada um desses protocolos a complexidade computacional em relação ao nosso método de ataque é diferenciada. A complexidade do protocolo Random-HB$^{\#}$ aumenta de forma linear, em m, em relação aos nossos ataques ao protocolo HB^{+}, uma vez que cada coluna das matrizes é gerada de forma aleatória. Esse pequeno aumento quanto a complexidade é insignificante em relação ao excessivo aumento de área de memória para se armazenar essas matrizes no *chip* RFID.

As chaves do protocolo HB$^{\#}$ são matrizes de Toeplitz, a s-ésima coluna corresponde ao deslocamento de uma posição em relação a anterior, sendo que a s-ésima coluna de cada uma das duas matrizes correspondem ao par de chaves do protocolo HB^{+}. Dessa forma podemos concluir que: a complexidade do protocolo HB$^{\#}$ em relação ao protocolo HB^{+} é praticamente a mesma, uma vez que ao se determinar a s-ésima chave, para se determinar a chave $(s+1)$-ésima faltam apenas dois bits, um da coluna corresponde a matriz X e o outro em relação a matriz Y. Essas $4(m-1)$ possibilidades podem ser facilmente definidas, de dois em dois bits, de forma prática, direta, e independente.

Portanto, a prova de segurança desses protocolos feita por seus autores, assim como a argumentação de que os dois protocolos têm nível de segurança equivalente, com a vantagem de que o protocolo HB$^{\#}$ tem um ganho otimizado de área de memória para armazenamento das matrizes X e Y, está equivocada. Consideramos aqui que as chaves do protocolo HB^{+} possuem tamanhos equivalente a cada par de chaves dos protocolos Random-HB$^{\#}$ e HB$^{\#}$.

3.13 Ataque ao Protocolo HB*

Duc e Kim [20] propuseram o protocolo HB*, versão alterada do protocolo HB^{+} [24], com o objetivo de torná-lo resistente a ataques ativos, como o *man-in-the-middle*, pressupondo que este é resistente a ataques passivos.

Neste protocolo, como ilustra a figura 3.5, as chaves $\{x, y, s\}$ têm comprimento k, $\{a, b\}$ são sequências aleatórias binárias com comprimento k, e $\{\gamma, \nu\}$ têm distribuição de Bernoulli com probabilidade η' e η, respectivamente.

Primeiramente, a etiqueta gera b e γ, calcula $w = (b \odot s) \oplus \gamma$, e envia $\{b, w\}$ para a leitora. A leitora retorna a sequência a. A etiqueta verifica se $\gamma = 0$, se afirmativo calcula $z = (a \odot x) \oplus (b \odot y) \oplus \nu$, caso contrário calcula $z = (b \odot x) \oplus (a \odot y) \oplus \nu$; e envia z para a leitora. A leitora verifica se $b \odot s = w$, caso afirmativo $\gamma = 0$, então verifica se z é igual a $(a \odot x) \oplus (b \odot y)$, ou em caso contrário, $\gamma = 1$, verifica se z é igual a $(b \odot x) \oplus (a \odot y)$.

O ataque a este protocolo é simples, pode ser feito tanto pelo nosso método probabilístico como pelo nosso método por Falha, com a diferença de este possuir duas etapas:

Etapa 1:

1. O atacante monitora a comunicação e armazena $\{b, w, a, z\}$,
2. Em separado monta m_k equações com $\{b, w\}$,
3. Escolhe aleatoriamente um sub-conjunto de equações entre as m_k possíveis, e resolve o sistema,
4. Em havendo solução, provavelmente deve ter determinado a chave s, deve-se checar esta solução com $\{b, w\}$. Caso afirmativo, vá para a Etapa 2; caso contrário vá para o item 3.

Etapa 2:

1. Uma vez conhecendo $\{b_i, w_i, s\}$, determina-se cada γ_i, de cada etapa,
2. O atacante monta m_{2k} equações, realizando a seguinte verificação: Se $\gamma_i = 0$, a equação é do tipo $a_i||b_i = z_i$, caso contrário é $b_i||a_i = z_i$,
3. Escolhe aleatoriamente um sub-conjunto de equações entre as m_{2k} possíveis, e resolve o sistema,
4. Em havendo solução, as chaves x e y foram determinadas, deve-se checar a solução com $\{a_i, b_i, z_i\}$, fim do processo; caso contrário vá para o item 3.

Portanto, a complexidade do ataque ao protocolo HB* aumenta em relação ao do protocolo HB$^+$ de forma linear, basicamente, os dois métodos têm a mesma complexidade. A diferença básica é que para se atacar o protocolo HB* deve-se fazê-lo em duas etapas, independentes. O ganho esperado pelos autores no aumento de complexidade do protocolo HB* não ocorreu.

3.14 Conclusões

Nós apresentamos um novo ataque passivo e probabilístico contra os protocolos HB e HB$^+$. Comparado aos ataques BKW e FMICM, o nosso método criptoanalítico proposto apresenta algumas vantagens:

1. Não requer pré-processamento,
2. Não requer prévio conhecimento do percentual de ruído,
3. A quantidade de pares desafios-resposta necessários para se quebrar os protocolos é drasticamente reduzido, fazendo com que o nosso método se torne mais viável para ser implementado.

O nosso método de quebra dos protocolos HB e HB$^+$ muda o enforque sobre os métodos de quebra já propostos na literatura para esse tipo de protocolo, não só porque utiliza-se de uma quantidade de amostras bastante reduzida como também é um método probabilístico. A escolha aleatória de um subconjunto de desafios, agregado a um tipo de método bem elaborado de decisão para determinação do resultado correto, é um excelente método para quebra desse protocolo.

Nós finalmente enfatizamos que o nosso ataque permite que se faça uma relação de conveniência entre o esforço computacional e a quantidade de pares desafios-resposta necessários para se quebrar os protocolos. Em princípio, reduzir o esforço computacional para quebrar os protocolos HB e HB$^+$ aumentando a quantidade de pares de desafios-resposta em função da quantidade disponível. É também possível quebrar esses protocolos com menos pares desafios-resposta, aumentando a complexidade computacional. O mesmo não é possível para os métodos BKW e FMICM, onde, em qualquer caso, a quantidade de pares desafios-resposta cresce exponencialmente com o comprimento da chave.

Por fim, nosso ataque pode ser aplicado também aos outros integrantes da família HB (diferentemente dos outros ataques previamente considerados na literatura).

Capítulo 4

Ataques por Falha aos protocolos HB, HB^{+}, $HB^{\#}$, e Random-$HB^{\#}$

4.1 Introdução

Considere um protocolo que estabeleça uma comunicação segura entre uma pessoa e um dispositivo eletrônico. Embora o meio de comunicação possa ser inseguro, a finalidade deste protocolo deve ser a de impossibilitar a ação de um atacante, que mesmo de posse do dispositivo, não tenha condições de extrair informação suficiente que lhe permita deduzir parcial ou totalmente a chave secreta embutida em área de memoria deste dispositivo, mesmo utilizando-se de quaisquer ações técnicas.

Um Ataque por Falha a algoritmo ou protocolo criptográfico consiste em assumir que o atacante está de posse do dispositivo criptográfico e tem condições de alterar a área de memória onde está armazenada a chave secreta, ou registradores internos ao algoritmo ou protocolo. Assume-se que o atacante não tem conhecimento prévio desta chave, mas pode trocar, provocar falha, em qualquer um dos seus bits (se for zero torna-se um, e vice-versa) que estão na memória RAM ou nos registradores internos ao dispositivo criptográfico. O atacante pode também sempre que necessário reinicializar o dispositivo criptográfico e aplicar novas falhas com o objetivo de extrair informação diferenciada sobre a chave ou sobre um determinado registrador, com o objetivo final de determinação da chave secreta. O método de Ataque por Falha a algoritmo criptográfico foi aplicado primeiramente por Boneh et al. [13] em algoritmos criptográficos assimétricos, e posteriormente por Hock e Shamir [31] em algoritmos simétricos.

Vários protocolos de identificação e autenticação têm sido criados, alguns deles têm como premissa de que os dispositivos onde são implementados devem ser confiáveis. Esta condição certamente pode ser uma barreira quanto à confiabilidade destes protocolos em relação à segurança criptográfica que oferecem no mundo real. Temos que levar em consideração a existência desses ataques por falha, embora o atacante não consiga ler a área de memória onde estão gravados os códigos e as chaves, mas conseguem alterar o seu conteúdo, sem conhecê-los.

Nós apresentaremos três métodos distintos de ataques por falha aos protocolos HB, HB^{+},

HB$^{\#}$, e Random-HB$^{\#}$. O ataque ao protocolo HB$^{\#}$ é um caso particular do ataque ao protocolo Random-HB$^{\#}$. Cada um dos ataques foi especialmente desenhado para estes protocolos, sendo bastante eficientes. Para se determinar a chave do protocolo HB, que pode ter milhares de bits, são necessários pouco mais que 24 pares de desafios-resposta, e provocar k falhas (erros) na área de memória onde estiver armazenada a chave. Para se determinar as chaves do protocolo HB^{+} são necessários $2k$ falhas. Para se determinar as chaves dos protocolos Random-HB$^{\#}$ e HB$^{\#}$ basta provocar no máximo $(k_x + k_y) \times (m + 2^{S-1} - S)$ falhas, onde S é o número mínimo de falhas simultâneas que devem ser provocadas em cada linha k_x e k_y das matrizes de chaves X e Y, respectivamente, de tal forma que a leitora deixe de autenticar a etiqueta. Esta variável S será melhor detalhada na seção 4.4.

Até a publicação de nossos resultados, ataques por falhas nunca haviam sido aplicados contra protocolos de identificação destinados a dispositivos RFID.

O restante desse capítulo está organizado da seguinte forma: na seção 4.2 apresentamos o nosso ataque por falha ao protocolo HB, e na seção 4.3 é apresentado um novo ataque ao HB^{+}. Na seção 4.4 apresentamos um ataque específico aos protocolos HB$^{\#}$ e Random-HB$^{\#}$, e na seção 4.5 apresentamos nossas conclusões deste capítulo.

4.2 Método de Ataque por Falha ao protocolo HB

O protocolo HB está definido na seção 2.3.1. Nós apresentaremos aqui uma proposta de ataque ativo para o protocolo HB considerando que existe uma chave secreta e compartilhada entre a leitora e cada etiqueta. Para que cada autenticação ocorra, são gerados r sequências de bits aleatórios $a_i \in \{0,1\}^k$.

Para a realização deste ataque pressupõe-se que o atacante está de posse do dispositivo criptográfico onde está implementado o protocolo HB. O nosso método de ataque por falha a este protocolo corresponde ao atacante alterar o j-ésimo bit x_j^t da chave x, em um dado instante t, sem o conhecimento prévio deste bit, com o objetivo de se determinar esta chave. Nós assumimos também que o atacante não tem conhecimento da probabilidade η de ruído. O ataque consiste na realização de algumas poucas etapas, em média 24, para chaves com até 1000 bits aproximadamente. O método proposto é bastante simples de ser implementado e corresponde aos seguintes procedimentos por parte do atacante:

Considere a relação $w_i = f(a_i, x) = a_i \odot x$, onde $\odot$ corresponde ao produto interno módulo 2.

1 Capture m pares de desafio-resposta $\{\mathbf{a}_i, z_i\}$ durante procedimentos de autenticação legítimos.

2 Envie por 10 vezes cada um desses desafios $\mathbf{a}_i$ para a leitora (ou dispositivo de autenticação) e armazene as 10 respostas correspondentes no vetor $\mathbf{z}_i^r$. Em média, $\mathbf{z}_i^r$ será igual a w_i em $10(1-\eta)$ vezes, e, usualmente, $0 < \eta \leq 0.40$. Então, w_i será o valor com maior ocorrência no vetor $\mathbf{z}_i^r$.

3 Para cada par $\{\mathbf{a}_i, w_i\}$, selecione um bit $\mathbf{a}_i^j$, tal que $\mathbf{a}_i^j = 1$, e inverta o valor deste bit. A sequência resultante é representada por $\mathbf{a}_i^*$, onde $\mathbf{a}_i^{j*} = \mathbf{a}_i^j \oplus 1$.

4 Envie por 10 vezes cada desafio modificado $\mathbf{a}_i^*$ para a leitora e armazene as 10 respostas correspondentes no vetor $\mathbf{z}_i^S$. Seja w_i^S o resultado de $f(\mathbf{a}_i^*, \mathbf{x}) = \mathbf{a}_i^* \odot \mathbf{x}$. Em média, $\mathbf{z}_i^S$ será igual a w_i^S em $10(1-\eta)$ vezes, e, usualmente, $0 < \eta \leq 0.40$. Então, w_i^S será o valor com maior ocorrência nesse vetor $\mathbf{z}_i^S$.

5 Se $w_i^S = 1$, retorne ao passo 3; ou

6 Provoque falha no j-ésimo bit da chave $\mathbf{x}^j$. A sequência resultante deve ser representada por $\mathbf{x}^*$, onde $\mathbf{x}^{j*} = \mathbf{x}^j \oplus 1$.

7 Envie por 10 vezes o desafio original $\mathbf{a}_i$ para a leitora e armazene as 10 respostas correspondentes no vetor $\mathbf{z}_i^F$. Seja w_i^F o resultado da função $f(\mathbf{a}_i, \mathbf{x}^*) = \mathbf{a}_i \odot \mathbf{x}^*$. Em média, $\mathbf{z}_i^F$ será igual a w_i^F em $10(1-\eta)$ vezes, e, usualmente, $0 < \eta \leq 0.40$. Então, w_i^F será o valor com maior ocorrência no vetor $\mathbf{z}_i^F$.

8 Se $w_i^S \neq w_i^F$, então $\mathbf{x}^j \leftarrow w_i$.

9 Se $j = k$ e ainda existirem bits desconhecidos na sequência $\mathbf{x}$, retorne ao passo 2 e utilize outra sequência $\mathbf{a}_i$.

Este nosso método de ataque é bastante geral e muito eficiente. Utilizando-se deste método, um atacante sem o conhecimento da probabilidade η de ruído pode recuperar todos os bits da chave secreta x aplicando algumas poucas falhas no dispositivo. Esta técnica apenas requer uma captura prévia de muito poucos pares de desafio-resposta gerados durante um procedimento de autenticação legítimo. É importante lembrar que essa quantidade de pares é bem menor do que qualquer outra técnica existente.

Apresentamos agora este método de ataque em pseudo-código. O Algoritmo 1 representa a função $Flip_i(y)$ que realiza a troca (provoca falha) no i-ésimo bit da sequência y:

Algorithm 1 FLIP$_\text{i}$($\mathbf{y}$)
Require: $\mathbf{y}$ tal que $\mathbf{y} \in \{0,1\}^n$
Ensure: $\mathbf{y}^*$, tal que $\mathbf{y}_i^* = \mathbf{y}_i \oplus 1$

O Algoritmo 2 calcula $w_i = f(\mathbf{a}_i, \mathbf{x}) = \mathbf{a}_i \odot \mathbf{x}$:

Algorithm 2 Calcula W($\mathbf{a}_i, \mathbf{x}$), $w_i = \mathbf{a}_i \odot \mathbf{x}$
Require: $\mathbf{a}_i$ tal que $\mathbf{y} \in \{0,1\}^k$
Ensure: $w_i = \mathbf{a}_i \odot \mathbf{x}$
  for $t = 1$ até 10 **do**
    Envie $\mathbf{a}_i$ para o dispositivo
    $\mathbf{z}_i^r(t) \leftarrow$ RESPOSTA($\mathbf{a}_i$)
  end for
  $w_i \leftarrow$ Maior ocorrência($\mathbf{z}_i^r$)
  return w_i

Com o Algoritmo 3 quebra-se o protocolo HB:

Algorithm 3 BreakHB($\{\mathbf{a}_i, z_i\}$): Determina a chave secreta e compartilhada $\mathbf{x}$.

Require: Coleção de m pares de desafios $\{\mathbf{a}_i, z_i\}$ gerados durante um procedimento legítimo de autenticação.

Ensure: $ExtraiChave \in \{0,1\}^k$ e $ExtraiChave = \mathbf{x}$ com alta probabilidade.

```
ExtraiBits ← 0
while ExtraiBits < k do
  for i = 1 to m do
    for j = 1 to k do
      if a_i^j = 1 then
        a_i^* ← FLIP_j(a_i)
        w_i^S ← Calcula-W(a_i^*, x)
        if w_i^S = 0 then
          x^* ← FLIP_j(x)
          w_i^F ← Calcula-W(a_i, x^*)
          if w_i^F ≠ w_i^S then
            ExtraiChave(j) ← Calcula − W(a_i, x)
            ExtraiBits ← ExtraiBits + 1
          end if
        end if
      end if
    end for
  end for
end while
return ExtraiChave
```

4.3 Método de Ataque por Falha ao protocolo HB$^+$

O protocolo HB$^+$ está definido na seção 2.3.2. Na seção 3.7 foi feita uma equivalência desse protocolo ao HB em relação ao nosso ataque passivo probabilístico. Nesta seção, nós apresentaremos uma proposta de ataque ativo ao protocolo HB$^+$ considerando que este protocolo possui duas chaves secretas e compartilhadas entre a leitora e cada etiqueta. Para que cada processo de autenticação ocorra, são gerados r pares de sequências de bits aleatórios $a_i, b_i \in \{0,1\}^k$.

Para a realização deste ataque pressupõe-se que o atacante está de posse do dispositivo criptográfico onde está implementado o protocolo HB$^+$. Nós assumimos que o atacante não tem conhecimento da probabilidade η de ruído.

O nosso método de ataque corresponde aos seguintes procedimentos: (i) altera-se apenas um bit de x, ou de y, para zero; (ii) se esse bit for originalmente zero, qualquer tentativa de autenticação posterior a falha resultará em erro próximo a η, i.e., a probabilidade de erro se manterá; (iii) se

o valor original do bit alterado for igual a 1, o erro ao se tentar realizar qualquer autenticação passará para próximo de 50%, i.e., não será realizada a autenticação. Através da repetição desse procedimento, identificamos todos os bits da chave secreta.

No caso especifico do protocolo HB$^+$ o ataque consiste em provocar $2k$ erros no *chip* da etiqueta, um por vez, referentes as chaves x e y, aqui chamadas apenas por chave w, onde $w = x||y \in \{0,1\}^{2k}$. Para cada bit da chave fixado como zero (pode ser um), no *chip*, o atacante deve tentar realizar uma nova autenticação. Se a Leitora autenticar a Etiqueta é porque o bit correto da chave é zero, caso contrário pode ser igual a a um. Repetindo-se o procedimento, obtemos certeza quanto ao valor do bit em questão. Abaixo descreveremos este nosso ataque representado pelo Algoritmo 4:

Algorithm 4 AtaquePorFalhaHB$^+$($\{\mathbf{a}_i, \mathbf{b}_i, z_i\}$): Determinam-se as chaves secretas e compartilhadas **x** e **y**, bit a bit.

Require: Provocam-se erros nas chaves x, e y, e tentam-se realizar procedimentos de autenticação.
Ensure: $ExtraiChave \in \{0,1\}^{2\times k}$ e $ExtraiChave = x||y$.

```
w ← x||y
for i = 1 to 2 × k do
    w_i ← 0
    Realizar um processo de Autenticação
    if Sucesso then
        ExtraiChave_i ← 0
    else
        ExtraiChave_i ← 1
        w_i ← 1
    end if
end for
return ExtraiChave
```

4.4 Ataque por Falha aos Protocolos HB$^\#$ e Random-HB$^\#$

Os protocolos HB$^\#$ e Random-HB$^\#$, definidos no seção 2.3.4. Eles utilizam-se de m chaves distintas e de um único par $\{a, b\} \in \{0,1\}^k$ de sequências aleatórias para realizar cada autenticação solicitada pelas etiquetas. Por sua vez, cada etiqueta retorna um vetor z com m bits.

O nosso método de ataque consiste em determinar a matriz X, e depois a matriz Y. Para facilitar a discussão e poder fazer referência ao acesso em área de memória do *chip* por parte do atacante, durante o ataque, e por motivo de conveniência de programação, o algoritmo 5 atribui a W ora o conteúdo de X, ora o de Y. Os bits das chaves ao serem definidos como resultado de cada análise são armazenados em uma única matriz com dimensão $(k_x + k_y) \times m$.

Nós propomos um ataque por falha bastante eficiente e prático, para cada etapa da análise este método de ataque altera S bits, $S > 1$, da área de memória onde estão as chaves. Este método de ataque implica em alterar pelo menos dois bits simultâneos da chave porque o erro para estes protocolos corresponde a inequação $Hwt(z) < t$. Em um primeiro momento, o nosso ataque

consiste em determinar o número mínimo de erros possíveis que devem ser provocados na chave de maneira que seja o suficiente para inverter o sinal da relação $Hwt(z) < t$, i.e., inverter para $Hwt(z) > t$, e o dispositivo deixar de fazer autenticação. Em um segundo momento, o ataque consiste em determinar todos os bits restantes dessa chave, um a um.

Seja S a quantidade de bits a serem alterados em S colunas de X, ou de Y, em cada iteração. Inicia-se a busca desse S mínimo a partir de $S = 2$. Caso a quantidade de erros estabelecida seja insuficiente, esta deve ser incrementada até que se determine o mínimo de erros necessários. Esta busca por determinar o valor de S ideal ocorre com o seguinte procedimento: (i) Escolhe-se de forma aleatória S posições distintas de memória, referentes aos bits da i-ésima linha de X, ou de Y, e armazene em G_S; (ii) Modificam-se os bits nessas posições para zero, i.e., $x_{i,G_j} = 0$, ou $y_{i,G_j} = 0$, para $j \in \{1, ..., S\}$. O nosso ataque consiste em alterar S bits que sejam originalmente iguais a "1", com o objetivo de se aumentar o erro $Hwt(z)$; (iii) Após provocar os S erros no *chip*, o atacante deve tentar verificar se a leitora continua autenticando a etiqueta; (iv) Se ocorrer de a leitora não autenticar a etiqueta é porque os S bits alterados para zero, originalmente eram todos iguais a um.

Após o atacante inserir S erros no *chip*, caso a leitora continue autenticando a etiqueta, pode ter ocorrido uma das seguintes situações: (i) pelo menos um dos bits x_{i,G_j} das S posições alteradas é originalmente igual a 0; ou (ii) a quantidade S de bits alterados ainda não é o suficiente para inverter a relação $Hwt(z) < t$. Em um primeiro instante, o atacante deve escolher outras S posições de memória e provocar outros S erros. Em média, deve-se repetir esse processo 2^{S-1} vezes. Depois disso, se este procedimento ainda não tiver obtido sucesso, o atacante deve incrementar S. Baseado nos resultados de simulação, a quantidade S de bits a serem alterados deve ser bastante pequena, próxima a cinco, ou até no máximo dez, dependendo dos tamanhos das chaves, k_x e k_y.

Ao se inverter a relação de erro para $Hwt(z) > t$, o atacante terá determinado a quantidade mínima de bits S. Por este motivo, em seguida, o atacante deve reverter apenas um dos S bits x_{i,G_*}, ou y_{i,G_*}, para "1", retornando-o ao valor original, e manter os outros $S - 1$ iguais a zero. Este procedimento tornará bastante eficiente o ataque, uma vez que o atacante não precisará fazer combinações de m elementos, S a S, para determinar os outros $m - S$ bits da chave $x_{i,*}$, ou $y_{i,*}$.

Como o ataque é progressivo, o objetivo primeiro do atacante deve ser o de determinar este número mínimo S de erros que inverta a relação $Hwt(z)$. Para tanto, o nosso método de ataque inicia-se com $S = 2$. O passo seguinte consiste em determinar os $m - S$ bits restantes da i-ésima linha de X, ou de Y. Este procedimento é mais simples: utilizando-se dos $S-1$ bits alterados para zero, altera-se um a um os $m - S$ bits restantes, i.e., diferentes de x_{i,G_*}, ou y_{i,G_*}. Em cada caso, verificar se a leitora continua autenticando a etiqueta. Caso continue, o bit $x_{i,*}$, ou $y_{i,*}$, é igual a zero, caso contrário é igual a um. A complexidade média deste ataque é $(k_x + k_y) \times (m + 2^{S-1} - S)$. Este ataque está descrito em pseudo-código pelos Algoritmos 5 e 6 da seguinte forma:

Algorithm 5 AtaquePorFalhaHB$^{\#}(\{\mathbf{a}, \mathbf{b}, Z\})$: Determinam-se as chaves secretas e compartilhadas $\mathbf{X}e\mathbf{Y}$.

Require: Provocam-se erros nas chaves X, e Y, e tentam-se realizar procedimentos de autenticação.

Ensure: $ExtraiChave \in \{0,1\}^{(k_x+k_y)\times m}$ e $ExtraiChave = X||Y$.

```
S ← 2
for L = 0 to 1 do
  if L = 0 then
    W = X
    K = k_x
  else
    W = Y
    K = k_y
  end if
  for i = 1 to K do
    Autenticacao ← Sim
    while Autenticacao = Sim do
      q ← 0
      while (q < 2^{S-1}) and (Autenticacao = Sim) do
        for j = 1 to S do
          G_j ← Random(*) ∈ 1, m
          W_{i,G_j} = 0
        end for
        Realizar um processo de Autenticação
        if Parou de Autenticar then
          for j = 1 to S do
            ExtraiChave_{(i+L×k_x),G_j} ← 1
          end for
          Autenticacao ← Nao
        else
          q ← q + 1
        end if
      end while
      if Autenticacao = Sim then
        S ← S + 1
      end if
    end while
    Escolher um G_j, j = 1, S
    DeterminaMaisBitsAtaquePorFalhaHB#({G_j, i, W, m, S, ExtraiChave})
  end for
end for
Z ← ExtraiChave
return Z
```

Algorithm 6 DeterminaMaisBitsAtaquePorFalhaHB$^{\#}$($\{G_j, \mathbf{i}, \mathbf{W}, \mathbf{m}, \mathbf{S}, \mathbf{ExtraiChave}\}$): Determinam-se os outros bits das chaves secretas e compartilhadas $\mathbf{X} e \mathbf{Y}$.

Require: Provocam-se erros em W (chaves X ou Y) e tentam-se realizar procedimentos de autenticação.

Ensure: $ExtraiChave \in \{0,1\}^{(k_x+k_y)\times m}$.

```
W_{i,G_j} ← 0
for n = 1 to m do
  if (n ≠ G_j) and (com j ∈ {1, ..., S}) then
    W_{i,n} ← 0
    Realizar Uma Autenticação
    if Sucesso then
      ExtraiChave_{(i+L×k_x),n} ← 0
    else
      ExtraiChave_{(i+L×k_x),n} ← 1
    end if
  end if
end for
return ExtraiChave
```

4.5 Conclusões

Primeiramente, para mostrar a eficiência deste nosso ataque por falha vamos comentar alguns dos resultados obtidos da literatura com ataques aos protocolos HB/HB$^+$. Aplicando-se o método BKW, Blum et al. [12], com chave de 128 bits e erro = 25%, o esforço computacional é 2^{56}, utilizando-se de 2^{56} amostras. Ao se aplicar o método proposto por Carrijo et al. [17], o esforço computacional é igual a 2^{54}, utilizando-se de 2^{13} amostras. Quando aumentamos o tamanho da chave para 256 bits, e mantemos o erro em = 25%, um pior caso, o esforço computacional, do primeiro método, é 2^{88}, utilizando-se de 2^{88} amostras. Ao se aplicar o método proposto por Carrijo et al. [17], o esforço computacional é igual a 2^{107}, porém utiliza-se de apenas 2^{14} amostras.

Os nossos ataques, baseados nos pressupostos de ataques por falha, são bastante robustos, rápidos e eficientes. No caso dos protocolos HB/HB$^+$ o ataque independe inclusive do percentual de erro do ruído. Este método de ataque para o protocolo HB está implementado em linguagem de programação 'C', sendo de fácil alteração para permitir quaisquer quantidades de bits da chave secreta. Se a Chave secreta tiver 256 bits são necessários apenas 20 desafios em média e aproximadamente 10 segundos de processamento em um computador de 1.8GHz para quebrá-la. Se a chave tiver 128 bits são necessários apenas 17 desafios em média. Se a chave tiver 1024 bits são necessários apenas 24 desafios, enquanto que para uma chave com 10000 bits são necessários 31 desafios. É possível perceber que mesmo com a quantidade de bits da chave aumentando bastante, a quantidade de desafios aumenta de forma muito lenta. Estes resultados práticos evidenciam a fragilidade deste protocolo contra nosso ataque, deixando-o vulnerável, para qualquer que seja a chave compartilhada.

Para se quebrar o protocolo HB$^+$ é necessário apenas um desafio e realizar apenas $2k$ falhas. Para se quebrar os protocolos HB$^\#$ e Random-HB$^\#$ são necessárias no máximo $(k_x + k_y) \times (m + 2^{S-1} - S)$ falhas. Os nossos métodos de ataque são práticos e muito eficientes, com a sua aplicação torna-se possível a extração da chave gravada no dispositivo criptográfico onde estiverem implementados os protocolos HB, HB$^+$, HB$^\#$, ou Random-HB$^\#$.

Em suma, foi mostrado neste capítulo que os protocolos da família HB são muito vulneráveis contra adversários capazes de inserir falhas nas etiquetas RFID.

Capítulo 5

Ataque Ativo a Protocolos Baseados no Problema da Soma de k Mínimos

5.1 Introdução

Neste capítulo nós apresentamos uma ataque ativo a protocolos de autenticação e identificação baseados no Problema da Soma de k Mínimos (*Sum of k Mins*), proposto por Hopper e Blum [33]. O nosso método de ataque pressupõe que o atacante tem em mão um dispositivo com este protocolo implementado e tem condições de gerar e escolher os desafios. Nessas condições nós determinamos toda a chave composta de k pares de números inteiros, módulo n. Essa proposta de ataque é bastante eficiente, independe se o ruído está incluído nos desafios, assim como o atacante não necessita ter o conhecimento da taxa de erro para realizar o ataque.

O nosso método de ataque aqui proposto evidencia uma fraqueza deste protocolo, mostrando que o Problema da Soma de k Mínimos não é uma boa opção a ser empregada em protocolos de autenticação e identificação.

5.2 Definição do problema da Soma de k Mínimos

O Problema da Soma de k Mínimos pode ser definido da seguinte forma:

1. $z = \{(x_1, y_1), ..., (x_k, y_k)\}$ é um conjunto de pares (x_i, y_i) de inteiros, módulo n,
2. v é um conjunto com m sequências aleatórias. Cada sequência tem n inteiros módulo 10, i.e., $v_j \in \{0, ..., 9\}^n$, $j = 1, m$, e , $k \log_{10} n \leq m < \begin{pmatrix} n \\ 2 \end{pmatrix}$,
3. $$\mathrm{u_j} = f(v_j, zi) = \sum_{i=1}^{k} \min\{v_j[x_i], v_j[y_j]\} \bmod 10; j = 1, m; u_j \in \{0, 1, ..., 9\} \qquad (5.1)$$

No protocolo proposto por Hopper e Blum [33], z é a chave secreta e compartilhada, com $2k \log_2 n$ bits, v_j é a sequência aleatória e u_j são os desafios. Neste protocolo pode ser implemen-

tado ruído nos desafios com uma taxa η de ocorrência. Uma descrição detalhada do protocolo é apresentada no capítulo 2.

Para o nosso ataque estabeleceremos como pressuposto que o atacante não tem conhecimento da chave secreta e compartilhada z, e também não precisa conhecer a priori a taxa de ruído η, caso tenha sido implementado.

5.3 Estimativas de ataques anteriores

Como estimativas de ataques anteriores a este protocolo, Hopper e Blum [33] comentam que as restrições impostas garantem a sua segurança. Por exemplo, se $m \geq \begin{pmatrix} n \\ 2 \end{pmatrix}$ é possível quebrá-lo determinando-se a chave z a partir da resolução de sistemas de equações lineares. Se $m < \begin{pmatrix} n \\ 2 \end{pmatrix}$ o melhor algoritmo conhecido tem complexidade $\begin{pmatrix} n(n-1)/2 \\ k/2 \end{pmatrix} > \begin{pmatrix} n \\ k \end{pmatrix}$, para $k > 3$.

Se neste protocolo for incluído ruído, com probabilidade de ocorrência η, e se o atacante tiver menos que $\begin{pmatrix} n \\ 2 \end{pmatrix}$ amostras, a complexidade será maior do que $\begin{pmatrix} n \\ k \end{pmatrix}$. Para $n = 900$ e $k = 12$ a complexidade do ataque indicada por Hopper e Blum [33] é maior do que $\begin{pmatrix} 900 \\ 12 \end{pmatrix} \approx 2^{89}$.

5.4 Proposta de Ataque

Com a aplicação do nosso método de ataque, o atacante determina toda a chave z deste protocolo com muito pouco esforço computacional. Os nossos resultados mostram que um protocolo baseado no Problema da Soma de k Mínimos não resiste a um ataque com poucas sequências v_j escolhidas, podendo deixar vulneráveis os dispositivos de autenticação e identificação que tiverem implementados protocolos baseados neste Problema.

O nosso ataque consiste basicamente na geração de sequências escolhidas v_j, no cálculo dos desafios u_j correspondentes, e nas análises comparativas destes resultados, determinando-se de imediato os números inteiros pertencentes a chave secreta. Nestas análises determinam-se os inteiros $\{x_j, y_j\} \in z$, não necessitando de se montar e resolver qualquer sistema de equação, é preciso apenas de se realizar análises comparativas. A complexidade do nosso ataque não se altera se os desafios u_j tiverem ruídos e o atacante tiver domínio sobre o ruído, i.e., conhecer o ruído. Por outro lado, se o atacante não tiver controle sobre o ruído e nem conhecer a taxa de ruído, os nossos critérios estabelecidos para se determinar a chave secreta quase que não se alteram, mantendo-se praticamente a mesma complexidade.

O ataque consiste de duas etapas: a primeira consiste em determinar quais são os $2k$ números inteiros x_j e $y_j \in z$ existentes na chave; a segunda parte consiste em determinar cada par $(x_j, y_j) = (y_j, x_j) \in z$. O método de ataque apresentado abaixo foi implementado em linguagem C, em ambiente Windows, e o pseudo-código não está otimizado com o objetivo de se melhor facilitar a

compreensão.

Os passos referentes ao nosso método de ataque não consideram este protocolo com uma taxa α de ruído. No entanto, se o protocolo tiver sido implementado com ruído, e o atacante não tiver o seu controle, é suficiente que se gere aproximadamente 10 desafios u_j para cada sequência v_j escolhida. Consequentemente, aproximadamente $10(1-\alpha)$ destes desafios u_j calculados a partir da mesma sequência v_j estarão corretos, não se alterando então a análise. A descrição do ataque ativo a protocolos baseados no problema da soma de k mínimos é descrito nas duas seguintes partes:

Parte 1

1. Escolher um número r, tal que $2 \leq r \leq n$
2. Fazer $v_{ij} = r$ para qualquer $i = 0, n-1$, e $j = 0$
3. Calcular u_j, fazer $T \leftarrow u_j$
4. Atribuir a n valores v_{ij} tal que:
   (a) $v_{ij} \leftarrow r$ se $i \neq j$, e
   (b) $v_{ij} \leftarrow r-1$ se $i = j$, para $i, j = 0, n-1$
5. Calcular u_j, e fazer $w_j = u_j$, $j = 0, n-1$
6. Se $u_j = T \Rightarrow j \notin z$
7. Se $u_j = T - S \ mod \ n \Rightarrow j$ tem S ocorrências em z

Parte 2

1. Atribuir a n valores $v_{ij} \leftarrow 0$ para $i, j = 0, n-1$
2. Para todo $w_i \neq T$, fazer $v_{ij} = r$ e $p = i+1$
   (a) Se $w_p \neq T$, atribuir $v_{pj} \leftarrow r-1$
   (b) Calcular u_p
   (c) Retornar ao item (2) até que p seja igual a $n-1$
3. Se $u_p = r-1$, o par $(i, p) = (p, i) \in z$, para quaisquer p

Para melhor detalhamento, verifique o apêndice III.

Pode ser observado nessas duas etapas que o nosso método de ataque é simples, consiste basicamente na geração e escolha de sequências v_j, no cálculo dos desafios u_j e na realização de análises

simples. A complexidade deste método de ataque é diretamente relacionada com a quantidade de sequências que devem ser geradas, que depende dos parâmetros n e k. Quando a razão n/k aumenta, a probabilidade de haver pelo menos uma repetição na chave z com $2k$ inteiros, em n possíveis, gerados aleatoriamente, decresce rapidamente. Por exemplo, para $n = 900$ e $k = 12$, para $x_i \in z$, a probabilidade $P(x_i = x_j) \approx 0.02$, para $i \neq j$. Quanto menor for a quantidade de números x_j repetidos na chave z maior é a complexidade do ataque, porque maior será a quantidade de números a serem analisados, próximo ou igual a $2k$. Vamos estimar a complexidade para este pior caso, não considerando repetição, e assim o nosso ataque passa a ser probabilístico em função de n e k.

O nosso ataque se divide em duas etapas: Para a primeira etapa são necessárias $n+1$ sequências v_j, com as quais se determinam os números inteiros que fazem parte da chave secreta z. A segunda etapa compreende dos seguintes passos: Pega-se um dos inteiros pertencentes a esta chave, definido na etapa 1, geram-se $n-1$ sequências v_j, e determina-se o par correspondente após análise comparativa; em seguida pega-se um outro inteiro, geram-se $n-3$ sequências v_j e determina-se também o par correspondente. O processo segue até faltarem os dois últimos pares. Pega-se um dos quatro números e geram-se 3 sequências v_j. Ao se determinar este par, restarão dois inteiros que formam o k-ésimo par. Portanto, a quantidade de sequências desta segunda etapa corresponde a soma das seguintes quantidades: $n-1$, $n-3$, $n-5$, ... , 3. Estas quantidades formam uma progressão aritmética de razão dois com $(n-1-3)/2+1$ elementos. Portanto, há necessidade de se gerar no máximo a seguinte quantidade de sequências escolhidas:

$$N_{Seq} = \left(\frac{3+n-1}{2}\right) \cdot \left(\frac{n-1-3}{2}+1\right) + n + 1 = \frac{(n+2)(n-2)}{4} + n + 1$$

$$= \frac{(n^2-4)}{4} + n + 1 = \frac{n^2}{4} + n \tag{5.2}$$

A chave z tem $2k$ números inteiros, esta quantidade pode ser menor ou igual a n, i.e., $2k \leq n$. Para generalizar, esta equação pode ser escrita como $N_{Seq} = k^2 + n$, uma vez que serão analisados os inteiros existentes na chave, e não todos os inteiros *mod* n.

5.5 Análise dos Resultados

O ataque proposto é eficiente. Com sua aplicação é possível determinar a chave secreta em muito poucos passos. Para comparação, vamos utilizar o exemplo dado na seção 5.3 para $n = 900$ e $k = 12$. Hopper e Blum [33] comentam que a complexidade deste protocolo com estes parâmetros é maior do que 2^{89}, onde a chave é determinada a partir da montagem e cálculo de sistemas de equações lineares.

Para entendimento do nosso método, faremos a seguinte abordagem: para $k = 12$, a chave secreta z contém 24 números inteiros, cada um deles módulo 900. O maior esforço computacional ocorrerá quando a chave z tiver 24 números distintos, com probabilidade próxima a 0.98 (se houver alguma repetição o ataque é ainda mais eficiente). Como os números inteiros da chave z servem como endereços para os desafios u_j, existem pelo menos $900 - 24 = 876$ destes endereços que não

serão utilizados para qualquer que seja a chave. Portanto, na primeira etapa do nosso método, geram-se $n+1 = 900+1 = 901$ sequências v_j, onde se calculam os respectivos desafios u_j, e a partir de análises comparativas determinam-se os inteiros que pertencem à chave. Na segunda etapa deve-se gerar no máximo $k^2 - 1 = 12^2 - 1 = 144 - 1 = 143$ sequências e calcular os respectivos desafios u_j, que possibilitam também determinar por comparação todos os pares $(x_i, y_i) \in z$. Nesse ataque, o total de desafios gerados e escolhidos é de $143 + 901 = 1044 \approx 2^{10}$, não havendo necessidade de se montar qualquer sistema de equações, apenas análises comparativas simples.

5.6 Conclusões

O ataque proposto determina toda a chave z deste protocolo baseado no problema da soma de k mínimos [33]. O esforço computacional exigido é bem pequeno, bastando-se gerar e escolher no máximo $N_{Seq} = k^2 + n$ sequências, calcular os desafios correspondentes, e determinar os inteiros pertencentes a chave z a partir de análises comparativas. As etapas do ataque são fáceis de se implementar e exigem bastante poucos pares de amostras (v_j, u_j) para determinação da chave secreta. Como exemplo comparativo, quando $n = 900$ e $k = 12$, pelos métodos já apresentados, é necessário um esforço computacional maior que 2^{89}, a partir de resolução de sistemas de equações com dimensão 900×901. Os nossos resultados são bastante expressivos, uma vez que precisamos de apenas $1044 \approx 2^{10}$ pares (v_j, u_j), com v_j escolhidos, determinando-se toda a chave apenas por análise comparativa simples. Os nossos resultados mostram que um protocolo baseado no problema da soma de k mínimos não resiste a um ataque com poucas sequências v_j escolhidas, podendo deixar vulneráveis os dispositivos de autenticação e identificação que tiverem implementados esses protocolos.

Capítulo 6

Ataque Passivo a Protocolos Baseados no Problema da Soma de K Mínimos

6.1 Introdução

Neste capítulo, propomos um método de ataque passivo a protocolos baseados no problema da soma de k mínimos, conforme definido na seção 2.3.6. Para determinados valores de k, o nosso método de ataque é mais eficiente do que ataques por força bruta.

A complexidade computacional do nosso método de ataque para este protocolo depende diretamente da quantidade k de pares de inteiros existentes na chave secreta e compartilhada, e também de n, quantidade de inteiros $\{0, 9\}^n$ em cada desafio. Cabe comentar novamente que a implementação e uso deste protocolo são baseados em n grande e k pequeno.

O ataque é passivo e pressupõe-se que seja de conhecimento do atacante os pares (v_j, u_j) - que são os aleatórios e os desafios utilizados em processos de autenticação. Também fazemos a comparação do esforço computacional exigido, para qualquer valor de k, entre o nosso método de ataque versus o ataque por força bruta, que tem complexidade igual a n^{2k}.

6.2 Proposta de um Novo Ataque Passivo ao Protocolo

O método de ataque que propomos neste capítulo para este protocolo depende diretamente da quantidade k de pares de inteiros existentes na chave secreta e compartilhada, e também de n, quantidade de inteiros $\{0, 9\}^n$ em cada desafio. Descreveremos a nossa proposta de ataque deste protocolo, passo-a-passo, para $k = 1$, 2, e 3. O ataque é passivo e pressupõe-se que é de conhecimento do atacante os pares (v_j, u_j) - que são os aleatórios e os desafios utilizados em um processo de autenticação.

6.3 Ataque Passivo para $k = 1$

Ao aplicar o nosso método de ataque a esse protocolo para qualquer n, para $k = 1$ e utilizando-se de dois pares de desafios-resposta, (v_j, u_j), é possível determinar x_1 ou y_1. Com 4 desafios determina-se a chave $z = \{(x_1, y_1)\}$.

Para o nosso ataque pressupõe-se que o atacante armazenou m pares (v_j, u_j) a partir de algum processo de autenticação válido. Embora sejam necessários apenas quatro pares para se determinar o par (x_1, y_1), alguns outros pares devem ser utilizados para confirmação da chave. O ataque proposto consiste em:

1. Criação de dois vetores $Chave[n]$ e $Chave_r[n]$, com as n posições iguais a zero
2. Para o par (v_j, u_j), para cada $v_{ji} \in v_j$ e $v_{ji} = u_j$, $i = 1, n$, $j = 1$ fazer:
   $Chave[$ *endereco de* $v_{ji} = u_j$ $] = 1$
3. Enquanto $j \leq m$
4. $j \leftarrow j + 1$
5. Com o par (v_j, u_j), para cada $v_{ji} \in v_j$ e $v_{ji} = u_j$, $i = 1, n$, fazer:
   $Chave_r[$ *endereco de* $v_{ji} = u_j$ $] = 1$
6. $Chave[i] \leftarrow Chave[i] \otimes Chave_r[i]$, para $i = 1, n$
7. Se o vetor resultante $Chave[n]$ mantiver mais de um índice indexado igual a 1, voltar ao item 3, (isso ocorre em muito poucos casos)
8. Neste passo, x_1 ou y_1 já está determinado. O único índice indexado igual a 1 em $Chave[n]$ está na posição x_1 ou na y_1, i.e., $Chave[x_1] = 1$ ou $Chave[y_1] = 1$. Portanto, neste instante, sabe-se que $Z = \{(x_1, ?)\}$, para este protocolo a chave $\{(x_j, y_j)\}$ equivale a $\{(y_j, x_j)\}$

   Observação: Em média, para este ataque, são necessários apenas dois pares (v_j, u_j) para se determinar x_1 (ou y_1). Executando-se este conjunto de etapas novamente determina-se o correspondente y_1 (ou x_1).

6.4 Ataque Passivo para $k = 2$

Este ataque aqui criado foi implementado para $k = 2$, é diferente do implementado para $k = 1$, porém tem o mesmo pressuposto: o atacante deve armazenar m pares (v_j, u_j). Aplicando-se este ataque para qualquer valor de n, são necessários em média apenas no máximo três destes pares para se determinar a chave $z = \{(x_1, y_1), (x_2, y_2)\}$. Ao se determinar um candidato à chave, parte dos outros pares armazenados deve ser utilizado para confirmação desta chave.

O método de ataque aqui proposto segue o seguinte raciocínio: cada u_j é determinado a partir do conteúdo de dois endereços de v_j, dos quatro existentes na chave, que são o $min\{v_j[x_1], v_j[y_1]\}$ e o $min\{v_j[x_2], v_j[y_2]\}$, somados módulo 10. A chave secreta pode ser determinada de forma

completa ao se analisar dois pares de endereços de v_j e de v_k cuja soma seja u_j e u_k, respectivamente, e também se estes dois pares contiverem os quatro índices $\{x_1, y_1, x_2, y_2\}$, em quaisquer combinações. Para tanto, dados quatro endereços candidatos, não é necessário permutá-los e analisá-los em todas as suas possibilidades, $(4! = 24)$. Basta utilizar-se das seguintes três permutações, que para este protocolo, equivalem às 24 permutações possíveis: $Z_1^* = \{(x_1^*, y_1^*), (x_2^*, y_2^*)\}$, $Z_2^* = \{(x_1^*, x_2^*), (y_1^*, y_2^*)\}$, e $Z_3^* = \{(x_1^*, y_2^*), (x_2^*, y_1^*)\}$. O nosso método de ataque para $k = 2$ segue os seguintes passos:

1. O atacante deve armazenar m sequências de aleatórios v_j e os correspondentes desafios u_j, $j = 1, m$. Em média são necessários apenas, e no máximo, 03 pares (v_j, u_j) para se determinar a chave. Os outros desafios devem ser utilizados para confirmação desta chave,

2. Para $v_{ji} \in v_j$, fazer todas as combinações dois a dois entre v_{ji} e v_{jk}, e calcular $u_j^* = v_{ji} + v_{jk} \bmod 10$. Armazenar os dois índices $w_{sj}^0 = i$ e $w_{sj}^1 = k$ destas combinações quando $u_j^* = u_j$. Para cada v_j existem r_j combinações iguais a u_j, então, $s = 1, r_j$

3. Em seguida, deve-se executar o algoritmo 7:

Algorithm 7 Ataque Passivo Soma de K Mínimos($\{v_j, u_j\}$): Determina a chave secreta $\mathbf{z}$, $k = 2$.

Require: Coleção de m pares de desafios $\{v_j, u_j\}$ gerados durante um procedimento legítimo de autenticação.

Ensure: $Z_a^* \in \{0,9\}^k$ e $Z_a^* = \mathbf{z}$.

for $j = 1$ to $m - 1$ **do**
  for $s = 1$ to r_j **do**
    $W_s \leftarrow \{w_{sj}^0, w_{sj}^1\}$
    for $k = j$ to m **do**
      for $t = s$ to r_k **do**
        $W_{st} \leftarrow \{w_{sj}^0, w_{sj}^1, w_{tk}^0, w_{tk}^1\}$
        Permutar W_{st} aos pares, são três possíveis
        $Z_1^* = \{(x_{11}^*, y_{11}^*), (x_{21}^*, y_{21}^*)\} = \{(w_{sj}^0, w_{sj}^1), (w_{tk}^0, w_{tk}^1)\}$, ou
        $Z_2^* = \{(x_{12}^*, y_{12}^*), (x_{22}^*, y_{22}^*)\} = \{(w_{sj}^0, w_{tk}^0), (w_{sj}^1, w_{tk}^1)\}$, ou
        $Z_3^* = \{(x_{13}^*, y_{13}^*), (x_{23}^*, y_{23}^*)\} = \{(w_{sj}^0, w_{tk}^1), (w_{sj}^1, w_{tk}^0)\}$
        for $a = 1$ to 3 **do**
          for $x = 1$ to m **do**
            $u_x^* = \sum_{i=1}^{k} \min\{v_x[x_{ia}^*], v_x[y_{ia}^*]\} \bmod 10$
            if $\mathbf{u}_x \neq \mathbf{u}_x^*$ **then**
              go to (for t)
            end if
          end for
          if $\mathbf{u}_x = \mathbf{u}_x^*$ **then**
            (para todo x) $\Rightarrow Z_a^* = \{(x_{1a}^*, y_{1a}^*), (x_{2a}^*, y_{2a}^*)\} = z$, FIM
          end if
        end for
      end for
    end for
  end for
end for
return Z_a^*

6.5 Ataque Passivo para $k = 3$

O ataque aqui criado e implementado para $k = 3$ é semelhante ao implementado para $k = 2$, algoritmo 7, onde o atacante deve armazenar m pares (v_j, u_j). Aplicando-se este ataque para qualquer valor de n, são necessários em média apenas e no máximo três destes pares para se determinar a chave $z = \{(x_1, y_1), (x_2, y_2), (x_3, y_3)\}$. Ao se determinar um candidato à chave, os outros pares armazenados devem ser utilizados para confirmação desta chave.

O método de ataque aqui proposto segue a seguinte idéia: cada u_j é determinado a partir do conteúdo de três endereços de v_j, dos seis existentes na chave, que são o $min\{v_j[x_1], v_j[y_1]\}$, o $min\{v_j[x_2], v_j[y_2]\}$ e o $min\{v_j[x_3], v_j[y_3]\}$, somados módulo 10. A chave secreta pode ser determi-

nada de forma completa ao se analisar pares de ternos de endereços de v_j e de v_k, cuja soma seja u_j e u_k, respectivamente, e se estes pares de ternos contiverem os seis índices $\{x_1, y_1, x_2, y_2, x_3, y_3\}$, em quaisquer combinações. Para tanto, dados seis endereços candidatos, não é necessário permutá-los e analisá-los em todas as suas possibilidades, $(6! = 720)$, mas sim basta utilizar-se de quinze permutações, que para este protocolo equivalem às 720 permutações possíveis, i.e., qualquer uma das 705 permutações restantes equivalem a uma destas quinze permutações, para este protocolo. O método de ataque aqui proposto para $k = 3$ segue os seguintes passos:

1. O atacante deve armazenar m desafios v_j e os correspondentes u_j, $j = 1, m$. Em média são necessários apenas 03 pares (v_j, u_j) para se determinar a chave. Parte dos outros desafios armazenados deve ser utilizada para confirmação desta chave. Esta quantidade m pode ser pequena, 10 a 20 pares é mais que o suficiente

2. Para $v_{ji} \in v_j$, fazer todas as combinações três a três entre v_{ji} , v_{jk}, e v_{jl} e calcular $u_j^* = v_{ji} + v_{jk} + v_{jl} \bmod 10$. Armazenar os três índices $w_{sj}^0 = i$, $w_{sj}^1 = k$ e $w_{sj}^2 = l$ destas combinações quando $u_j^* = u_j$. Para cada v_j existem r_j combinações iguais a u_j, então $s = 1, r_j$

3. Em seguida executar o algoritmo 8:

Algorithm 8 Ataque Passivo Soma de K Mínimos($\{v_j, u_j\}$): Determina a chave secreta $\mathbf{z}$, $k = 3$.

Require: Coleção de m pares de desafios $\{v_j, u_j\}$ gerados durante procedimentos legítimos de autenticação.

Ensure: $Z_a^* \in \{0,9\}^k$ e $Z_a^* = \mathbf{z}$.

for $j = 1$ to $m - 1$ **do**
  for $s = 1$ to r_j **do**
    $W_s \leftarrow \{w_{sj}^0, w_{sj}^1, w_{sj}^2\}$
    for $k = j$ to m **do**
      for $t = s$ to r_k **do**
        $W_{st} \leftarrow \{w_{sj}^0, w_{sj}^1, w_{sj}^2, w_{tk}^0, w_{tk}^1, w_{tk}^2\}$
        Permutar W_{st} aos pares, são quinze possíveis
        $Z_1^* = \{(x_1^*, y_1^*), (x_2^*, y_2^*), (x_3^*, y_3^*)\} = \{(w_{sj}^0, w_{sj}^1), (w_{sj}^2, w_{tk}^0), (w_{tk}^1, w_{tk}^2)\}$, ou
        $Z_2^* = \{(x_1^*, y_1^*), (x_2^*, x_3^*), (y_2^*, y_3^*)\} = \{(w_{sj}^0, w_{sj}^1), (w_{sj}^2, w_{tk}^1), (w_{tk}^0, w_{tk}^2)\}$, ou
        $Z_3^* = \{(x_1^*, y_1^*), (x_2^*, y_3^*), (x_3^*, y_2^*)\} = \{(w_{sj}^0, w_{sj}^1), (w_{sj}^2, w_{tk}^2), (w_{tk}^1, w_{tk}^0)\}$, ou
        $Z_4^* = \{(x_1^*, x_2^*), (y_1^*, y_2^*), (x_3^*, y_3^*)\} = \{(w_{sj}^0, w_{sj}^2), (w_{sj}^1, w_{tk}^0), (w_{tk}^1, w_{tk}^2)\}$, ou
        $Z_5^* = \{(x_1^*, x_2^*), (y_1^*, x_3^*), (y_2^*, y_3^*)\} = \{(w_{sj}^0, w_{sj}^2), (w_{sj}^1, w_{tk}^1), (w_{tk}^0, w_{tk}^2)\}$, ou
        $Z_6^* = \{(x_1^*, x_2^*), (y_1^*, y_3^*), (x_3^*, y_2^*)\} = \{(w_{sj}^0, w_{sj}^2), (w_{sj}^1, w_{tk}^2), (w_{tk}^1, w_{tk}^0)\}$, ou
        $Z_7^* = \{(x_1^*, y_2^*), (x_2^*, y_1^*), (x_3^*, y_3^*)\} = \{(w_{sj}^0, w_{tk}^0), (w_{sj}^2, w_{sj}^1), (w_{tk}^1, w_{tk}^2)\}$, ou
        $Z_8^* = \{(x_1^*, y_2^*), (x_2^*, x_3^*), (y_1^*, y_3^*)\} = \{(w_{sj}^0, w_{tk}^0), (w_{sj}^2, w_{tk}^1), (w_{sj}^1, w_{tk}^2)\}$, ou
        $Z_9^* = \{(x_1^*, y_2^*), (x_2^*, y_3^*), (y_1^*, x_3^*)\} = \{(w_{sj}^0, w_{tk}^0), (w_{sj}^2, w_{tk}^2), (w_{sj}^1, w_{tk}^1)\}$, ou
        $Z_{10}^* = \{(x_1^*, x_3^*), (x_2^*, y_2^*), (y_1^*, y_3^*)\} = \{(w_{sj}^0, w_{tk}^1), (w_{sj}^2, w_{tk}^0), (w_{sj}^1, w_{tk}^2)\}$, ou
        $Z_{11}^* = \{(x_1^*, x_3^*), (x_2^*, y_1^*), (y_2^*, y_3^*)\} = \{(w_{sj}^0, w_{tk}^1), (w_{sj}^2, w_{sj}^1), (w_{tk}^0, w_{tk}^2)\}$, ou
        $Z_{12}^* = \{(x_1^*, x_3^*), (x_2^*, y_3^*), (y_2^*, y_1^*)\} = \{(w_{sj}^0, w_{tk}^1), (w_{sj}^2, w_{tk}^2), (w_{tk}^0, w_{sj}^1)\}$, ou
        $Z_{13}^* = \{(x_1^*, y_3^*), (x_2^*, y_2^*), (x_3^*, y_1^*)\} = \{(w_{sj}^0, w_{tk}^2), (w_{sj}^2, w_{tk}^0), (w_{tk}^1, w_{sj}^1)\}$, ou
        $Z_{14}^* = \{(x_1^*, y_3^*), (x_2^*, x_3^*), (y_2^*, y_1^*)\} = \{(w_{sj}^0, w_{tk}^2), (w_{sj}^2, w_{tk}^1), (w_{tk}^0, w_{sj}^1)\}$, ou
        $Z_{15}^* = \{(x_1^*, y_3^*), (x_2^*, y_1^*), (x_3^*, y_2^*)\} = \{(w_{sj}^0, w_{tk}^2), (w_{sj}^2, w_{sj}^1), (w_{tk}^1, w_{tk}^0)\}$
        for $a = 1$ to 15 **do**
          for $x = 1$ to m **do**
            $u_x^* = \sum_{i=1}^{k} \min\{v_x[x_{ia}^*], v_x[y_{ia}^*]\} \bmod 10$
            if $\mathbf{u}_x \neq \mathbf{u}_x^*$ **then**
              go to (for t)
            end if
          end for
          if $\mathbf{u}_x = \mathbf{u}_x^*$ **then**
            (para todo x) $\Rightarrow Z_a^* = \{(x_{1a}^*, y_{1a}^*), (x_{2a}^*, y_{2a}^*), (x_{3a}^*, y_{3a}^*)\} = z$, FIM
          end if
        end for
      end for
    end for
  end for
end for
return Z_a^*

6.6 Ataque Passivo para qualquer valor de k

O nosso ataque passivo para um valor de k qualquer é semelhante ao implementado para $k = 2$ e 3, onde o atacante deve armazenar m pares (v_j, u_j). Aplicando-se este ataque para qualquer valor de n, são necessários em média poucos destes pares para se determinar a chave $z = \{(x_1, y_1), (x_2, y_2), ..., (x_k, y_k)\}$. Ao se determinar um candidato à chave, parte dos outros pares armazenados devem ser utilizados para confirmação desta chave.

O método de ataque aqui proposto segue a seguinte idéia: cada u_j é determinado a partir do conteúdo de k endereços de v_j, dos $2 \times k$ existentes na chave, que são o $min\{v_j[x_i], v_j[y_i]\}$, para $i = 1, k$, somados módulo 10. A chave secreta pode ser determinada de forma completa ao se analisar pares de $k - upla$ de endereços de v_j e de v_m cuja soma seja u_j e u_m, respectivamente, e se estes pares de ternos contiverem os $2 \times k$ índices $\{x_1, y_1, x_2, y_2, ..., x_k, y_k\}$, em quaisquer combinações. Para tanto, dados $2 \times k$ endereços candidatos, não é necessário permutá-los e analisá-los em todas as suas possibilidades, $(2 \times k)!$, mas sim basta utilizar-se de apenas das permutações que para este protocolo equivalem às $(2 \times k)!$ permutações possíveis. Para cada valor de k, deve-se determinar quantas são estas permutações. A implementação do nosso método de ataque é equivalente à implementação feita para $k = 3$, algoritmo 8, onde a variável W_{st} tem $2 \times k$ endereços. O número de permutações a ser definido para qualquer valor de k é:

$$\frac{(2k)!}{(2k) \cdot (2k-2) \cdot (2k-4) \cdots 2} = (2k-1) \cdot (2k-3) \cdots 3 \tag{6.1}$$

6.7 Relação Comparativa entre Complexidades: Método Exaustivo e o nosso Ataque Passivo

Em um ataque por exaustão, o esforço computacional para se determinar a chave z deste protocolo é de n^{2k}. No entanto, o atacante poderá considerar o fato de este protocolo possuir chaves equivalentes e realizar uma pré-análise, antes de exaurir todas as possíveis chaves indistintamente. Em princípio, este procedimento de se realizar uma triagem prévia de possíveis chaves pode ser considerado como um trabalho criptoanalítico, deixando assim de ser um ataque puramente exaustivo. As nossas análises e gráficos foram feitos baseados na realização dessa pré-análise quando se utiliza um método exaustivo, diminuindo o nosso ganho em desempenho quando comparamos estes dois ataques. Para valores bem pequenos de n e k foram feitas algumas verificações, implementadas em linguagem C, e por simulação foram obtidos resultados com valores maiores para n e k. A quantidade de chaves distintas, quando n e k crescem, tende a $n^{2k} \times 2^{-2k+1}$. Pode ser entendido que 2^{-2k+1} é uma taxa de redução das chaves equivalentes, entre todas as possíveis chaves.

Como comentado anteriormente, o nosso método proposto faz comparações a partir de pares de aleatórios e desafios. Para tanto, faz-se todas as combinações n, k a k de cada aleatório com n inteiros entre zero e nove (10 valores). Em média, $\frac{\binom{n}{k}}{10}$ destas combinações têm resultado igual ao desafio que está sendo analisado.

O esforço computacional para se analisar cada uma destas combinações, permutando-as, é

$$\prod_{i=1}^{k-1}(2k-2i+1) = \frac{(2k)!}{(2k)\times(2k-2)\times(2k-4)\cdots 2} = (2k-1)\times(2k-3)\cdots 3 \tag{6.2}$$

Portanto, o esforço computacional ao se fazer todas as comparações entre os resultados dos dois pares de aleatórios, incluindo as permutações é dado por:

$$\left[\frac{\binom{n}{k}}{10}\right]^2 \prod_{i=1}^{k-1}(2k-2i+1) \tag{6.3}$$

Foram feitos em Matlab gráficos da razão entre o esforço computacional exigido em um ataque por exaustão, já realizando triagem das chaves semelhantes, excluindo-as, e do esforço computacional necessário pelo método que propusemos, resultando na razão:

$$\frac{n^{2k}\cdot 2^{-2k+1}}{\left[\frac{\binom{n}{k}}{10}\right]^2 \prod_{i=1}^{k-1}(2k-2i+1)} = \frac{100\cdot n^{2k}\cdot 2^{-2k+1}\cdot (k!)^2}{\left[\prod_{i=0}^{k-1}(n-i)\right]^2 \cdot \prod_{i=1}^{k-1}(2k-2i+1)} \tag{6.4}$$

No Apêndice II são plotados 5 gráficos comparativos fazendo uma relação gráfica entre a complexidade computacional exigida pelo Método Exaustivo e pelo nosso Ataque Passivo: para $n = 32$ e $k = 1, 11$, figura II.1(a); para $n = 64$ e $k = 1, 13$, figura II.1(b); para $n = 128$ e $k = 1, 14$, figura II.2(a); para $n = 512$ e $k = 1, 16$, figura II.2(b); para $n = 1024$ e $k = 1, 17$, figura II.3(a).

6.8 Análise Comparativa entre o nosso método de Ataque Passivo e o por Exaustão

Observando-se os gráficos do Apêndice II é possível concluir que:

Para valores de k entre 5 e 8, o esforço computacional tanto do ataque por exaustão quanto o do nosso método proposto são praticamente equivalentes.

Para valores de k menores do que 5, o nosso método de ataque tem uma eficiência ligeiramente melhor, embora os protocolos dificilmente sejam implementados com parâmetros tão pequenos, por diminuir em muito a segurança.

Para valores de k maiores do que 8, quanto maior for k, muito mais eficiente é o nosso ataque comparado a um ataque por exaustão.

6.9 Conclusões

Baseado nos ataques anteriormente feitos para $k = 1, 2, 3$, assim como para sua extensão com qualquer valor para k, e observando-se também os gráficos, do Apêndice II, podemos verificar que o nosso método de ataque passivo a protocolos baseados no problema da soma de k mínimos é mais eficiente do que o por exaustão. O nosso método de ataque é simples e eficiente, mesmo considerando um método exaustivo onde se realiza a priori a exclusão de permutações equivalentes.

Capítulo 7

Ataques Passivo e Ativo por Falha ao Protocolo HB-MP

7.1 Introdução

Neste capítulo apresentamos dois métodos de ataque para o protocolo HB-MP, proposto por Munilla e Peinado [51]. O Protocolo HB-MP é um variante dos protocolos HB/HB$^+$, proposto por Hopper e Blum [32]. Um dos ataques propostos é passivo, enquanto o segundo método proposto é ativo.

Para o nosso método de ataque passivo, o atacante não precisa ter o controle da etiqueta com o chip RFID, mas necessita coletar e armazenar os desafios correspondentes a processos de autenticação válidos. Este método de ataque consiste em exaurir uma das duas chaves utilizadas no protocolo, e utilizando-se dos pares de desafios armazenados determinar todos os bits da outra chave. A complexidade computacional deste ataque é de $2^m + 2 \times (r - 1) \approx 2^m$, onde m é a quantidade de bits de cada desafio e de cada chave efetivamente utilizada, e r é a quantidade de passos de autenticação utilizados pelo protocolo. Para comparação, em um ataque por exaustão, a complexidade deste protocolo equivale a $2^{m+2\times r-1}$.

O segundo método de ataque se assemelha um pouco aos ataques por des-sincronização, por utilizarem também da prática de re-envio de desafios para a etiqueta, estes gerados pela leitora, em processos de autenticação válidos. Por outro lado, também se assemelha um pouco aos ataques por falha, por se utilizarem do método de troca do conteúdo de determinadas posições de registradores ou chaves em área de memória protegida. O atacante não tem conhecimento prévio do valor do bit naquela posição a ser alterada, o bit é alterado, em sua posição é gravado um bit igual a zero, ou igual a um. Em seguida o atacante realiza uma análise simples a partir dos resultados anteriores e posteriores a cada alteração.

O novo método de ataque altera um bit de cada vez, em cada desafio gerado pela leitora, em determinadas posições específicas, sob restrições previamente estabelecidas pelo atacante. Durante o ataque, o desafio alterado em um bit é re-enviado para a leitora para testar a restrição imposta pelo protocolo a partir das novas tentativas de autenticação. O ataque proposto difere destes outros

ataques também porque pressupõe a re-inserção na etiqueta ou na leitora dos desafios gerados pela própria etiqueta, tentando simular um processo de autenticação válido a partir de um desafio alterado.

7.2 Notação e Definição das Variáveis Utilizadas no Protocolo HB-MP

O Protocolo HB-MP compreende basicamente de r etapas onde a leitora e a etiqueta trocam desafios entre si, mantendo-se de forma secreta e compartilhada um par de chaves. Para cada desafio gerado pela leitora e enviado para a etiqueta, esta computa o resultado e gera um outro desafio correspondente que é enviado para a leitora. Descreveremos a seguir a notação e definição das variáveis utilizadas no Protocolo HB-MP:

1. A leitora e a etiqueta RFID compartilham as chaves secretas x e y, com k bits,
2. As sequências a_j^u, com m bits, são geradas pela leitora e enviadas para a etiqueta, j representa a j-ésima etapa da u-ésima autenticação,
3. As sequências b_j^u, com m bits, são geradas pela etiqueta e enviadas para a leitora, j representa a j-ésima etapa da u-ésima autenticação,
4. x_i , y_i, para $i = 0, k-1$, representam o i-ésimo bit de x e y, respectivamente,
5. O comprimento das mensagens trocadas entre a leitora e a etiqueta é m,
6. a_{ji}^u e b_{ji}^u, para $i = 0, m-1$, representam o i-ésimo bit de a_j^u e b_j^u, respectivamente,
7. Na j-ésima etapa deste protocolo, o bit y_j é utilizado para deslocar a chave x em uma posição, de forma circular para a esquerda, se $y_i = 1 \Rightarrow x = rotate(x, y_i)$,
8. $w \odot x$ representa o produto interno entre w e x,
9. ν_j é um bit de ruído gerado pela etiqueta, $\nu_j = 1$ tem probabilidade $\eta \in [0, 1/2)$,
10. $w \oplus \nu_j$ representa o *xor* entre w e ν_j,
11. x_m representa os m-ésimos bits menos significativos da chave x, $m \leq k$,
12. A quantidade r de etapas para a leitora autenticar a etiqueta deve variar conforme for a taxa de ruído η. Não há quantidades pré-fixadas, tanto r quanto η são pré-fixadas ao se implementar o protocolo,
13. Em um processo de autenticação, na etapa $j+1$ a chave x_m é diferente da chave na etapa j, se $0 \leq j \leq r-1$ e $y_j = 1$. Além disso, todas as sub-chaves terão os bits inseridos e ainda não utilizados se $r < k - m$, mesmo que $y_j = 1$, para qualquer j (pior caso),
14. $a_j \otimes b_j$ representa a operação *and*,

7.3 Descrição do Protocolo HB-MP

O protocolo HB-MP é aqui representado em sua j-ésima etapa e u-ésima autenticação. Uma vez fixado o número de etapas r para a leitora autenticar a etiqueta, os quatro passos devem ser executados r vezes:

Protocolo HB-MP, j-ésima etapa de r etapas:

1. A leitora gera uma sequência aleatória a_j^u, com m bits, e a envia para a etiqueta,
2. A etiqueta executa os seguintes comandos:
   (a) $x = rotate(x, y_j)$,
   (b) Gera o bit ν_j,
   (c) $z = x_m \odot a_j^u \oplus v_j$,
   (d) Geram-se b_j^u até que $a_j^u \odot x_m \oplus v_j = b_j^u \odot x_m$,
   (e) Envia b_j^u para a leitora.
3. Ao receber b_j^u da etiqueta, a leitora executa os seguintes comandos:
   (a) $x = rotate(x, y_i)$,
   (b) Se $a_j^u \odot x_m = b_j^u \odot x_m$, soma ao contador uma unidade.
4. Retorna ao item 1 por r etapas. Ao final, a leitora autentica a etiqueta caso o contador esteja próximo a $(1 - \eta)r$.

7.4 Método de Ataque I ao Protocolo HB-MP

Utilizando-se do método de ataque passivo, pressupondo que o protocolo HB-MP foi desenhado com duas chaves x e y com k bits cada uma, a complexidade computacional deste protocolo é igual a 2^{2k}. Como é realizado deslocamento circular na chave x, em função da chave y, em r etapas, até que a leitora autentique a etiqueta, podemos concluir que são utilizados r bits da chave y, e $m + r - 1$ bits da chave x, se e somente se $r \leq k - m$; este é o pior caso, considerando que em cada etapa são utilizados os m bits menos significativos da chave x. A complexidade computacional, neste caso, é reduzida para 2^{m+2r-1}. Empregando o nosso método de ataque esta complexidade pode ser reduzida ainda mais para $2^m + 2(r - 1) \approx 2^m$.

Este nosso ataque pode ser executado em dois procedimentos distintos: no primeiro, o atacante gera 2^m autenticações, em computadores em separado, armazenando todos os pares de desafios que serão utilizados para testes comparativos com $2r$ autenticações realizadas entre a leitora e a etiqueta. Neste caso, o atacante obtém solução única, porém há como dificuldade a coleta/geração de todos os desafios. No segundo procedimento, o ataque consiste em gerar muito menos autenticações, por exemplo $2r$, tornando bem mais viável o ataque, com o prejuízo de talvez se obter

mais de uma solução nas primeiras fases do ataque, resultando ao final em apenas uma solução. O nosso ataque, abaixo descrito, utiliza-se de $2r$ autenticações, tornando viável do ponto de vista do número de autenticações necessárias para se quebrar este protocolo. Este ataque consiste nas seguintes fases:

Método de Ataque I

1. Geram-se $2r$ autenticações, armazenando-se os desafios (a_j^u, b_j^u), para $j = 0, n-1$ e $u = 0, 2r-1$
2. **FASE I**
   (a) Faz-se exaustão nos m bits da chave x_m, utilizando-se apenas do primeiro par de desafios de cada autenticação realizada e armazenada pelo atacante, i.e., (a_0^u, b_0^u)
   (b) Verifica-se se $a_0^u \odot x_m = b_0^u \odot x_m$, para todo u. Em sendo verdadeira a igualdade, deve-se incrementar o contador
   (c) Selecionar a chave x_m cujo contador resultar em algo próximo a $(1-\eta) \times (2r-1)$. Neste momento, se houver mais de uma chave candidata, estas devem ser selecionadas.
3. **FASE II**
   (a) Para $j = 1, r-1$ (para se determinar os $r-1$ bits restantes)
   (b) Utilizar-se da chave x_m, item (c) FASE I
   (c) A partir de (a_j^u, b_j^u), verifica-se se $a_j^u \odot x_m = b_j^u \odot x_m$, para todo u. Em sendo verdadeira a igualdade, deve-se incrementar o contador
   (d) Seleciona-se a chave x_m cujo contador resulta em algo próximo a $(1-\eta) \times (2r-1)$
   (e) Se houver candidato $y_j = 0$
   (f) Caso contrário
   Deslocar x_m em uma posição à esquerda, $x_m = [x_{m-2}, x_{m-3}, ..., x_0, 0]$
   Para $\alpha_j = 0, 1$ (exaurir o bit menos significativo para x_m)
      i. Fazer $x_m = [x_{m-2}, x_{m-3}, ..., x_0, \alpha_j]$
      ii. A partir de (a_j^u, b_j^u), verificar se $a_j^u \odot x_m = b_j^u \odot x_m$, para todo u. Em sendo verdadeira a igualdade, deve-se incrementar o contador
      iii. Selecionar a chave x_m cujo contador resulta em algo próximo a $(1-\eta) \times (2r-1)$. Se houver candidato $y_j = 1$, e α_j é o $(m+j)$-ésimo bit de x.

7.5 Considerações sobre o Método de Ataque II ao Protocolo HB-MP

Neste segundo método de ataque ativo, ataque II, o atacante deve coletar Γ desafios, em média, referentes a processos de autenticação válidos. O parâmetro Γ depende do comprimento da chave x_m. Os desafios devem ser armazenados em duas matrizes A e B, com dimensão $\Gamma \times r$. Os elementos

$a_j^u \in A$ e $b_j^u \in B$ representam os desafios da j-ésima etapa e u-ésima autenticação, gerados e enviados pela leitora e pela etiqueta, respectivamente. A Tabela 7.1 indica a quantidade média de autenticações necessária para se determinar o par de chaves secreto, em função do comprimento da chave x:

Tabela 7.1: Quantidade média de autenticações necessárias para quebra do HB-MP

Comprimento m da chave x	128	256	512	1024	2048	8192
Γ	19	21	24	26	29	33

Para realização de uma autenticação, os desafios a_j^u e b_j^u são gerados de maneira aleatória. Utilizando-se do nosso método de ataque, o atacante fará uso apenas dos bits iguais a "1" de cada desafio. Suponha-se que esses possuem comprimento de 256 bits, conforme Tabela 7.6 serão necessários em média 21 autenticações para que todas as posições sejam analisadas.

O atacante deve também re-enviar para a etiqueta os desafios a_j^u coletados e alterados em um bit por vez, resultando em $a_j^{'u}$, conforme o nosso método de ataque, indicado na seção 7.6. Dado $a_j^{'u}$, o atacante deve inserir na etiqueta o desafio armazenado b_j^u correspondente, e certificar-se para que este seja o primeiro candidato para teste da restrição $a_j^{'u} \odot x_m \oplus v_j = b_j^u \odot x_m$.

7.6 Método de Ataque II ao Protocolo HB-MP

O nosso segundo método de ataque ao protocolo HB-MP consiste em determinar as chaves x e y compartilhadas entre a leitora e a etiqueta RFID. Em um processo de autenticação, as r etapas deste protocolo são realizadas em função dessas chaves, de tal forma que: (i) cada chave x_m (composta pelos m bits menos significativos de x) pode ser diferente em cada etapa da autenticação, uma vez que a chave y possibilita o deslocamento circular a esquerda da chave x; (ii) entre a etapa j e a etapa $j+1$ a chave x_m pode ser a mesma, caso $y_j = 0$; (iii) são utilizados no máximo $m+r-1$ bits da chave x, e apenas r bits da chave y; (iv) uma chave x_m será reiniciada a partir de x se e somente se $t \geq k - m$, sendo $t = \sum_{j=0}^{r-1} y_j$.

O nosso método de ataque pode ser dividido em $r+1$ etapas. As r primeiras etapas têm o mesmo procedimento de análise, esta parte do ataque será apresentada na seção 7.6.1. A última etapa do ataque, a etapa $r+1$, a ser apresentada na seção 7.6.2, consiste basicamente em analisar se a chave x foi ou não deslocada entre a etapa j e a $j+1$, e consequentemente tornando possível determinar o bit y_j correspondente a chave y. Nas primeiras r etapas, realizadas de forma independente, determinam-se todos os $m + r - 1$ bits de x, utilizados pela leitora e pela etiqueta, em qualquer processo de autenticação de etiqueta. Para tanto são utilizados $\Gamma \times r$ pares de desafios (a_j^u, b_j^u),

correspondentes aos Γ processos completos de autenticação armazenados pelo atacante, conforme Tabela 7.1.

Inicialmente, este nosso ataque consiste em determinar os m bits menos significativos da chave x, os x_m bits, utilizando-se apenas dos Γ pares (a_0^u, b_0^u). Em seguida, os $r-1$ bits restantes são determinados com os outros pares (a_j^u, b_j^u), para $j = 1, r-1$. Esta análise é semelhante à realizada para $j = 0$. Em cada uma das $r-1$ etapas deste ataque tem-se como objetivo o de determinar um bit de x, o bit menos significativo. Na implementação do nosso ataque, assim como no pseudo-código referente ao algoritmo 9, seção 7.6.1, nós optamos por determinar todos os m bits de x em cada etapa com a finalidade de uniformizar o sistema de ataque, uma vez que a complexidade computacional é irrelevante. Este procedimento deixa o sistema de ataque independente, sem qualquer interferência subjetiva por parte do atacante, ou do operador do sistema de ataque. Na implementação deste protocolo, se $k - m > r$, implica que $k - m - r + 1$ bits da chave x podem ser desconsiderados, porque nunca serão utilizados, assim como os $k - r$ bits da chave y.

7.6.1 Primeiras r etapas do Método de Ataque II

Na primeira parte deste ataque determinam-se os m bits menos significativos da chave x, a partir dos Γ pares de desafios (a_0^u, b_0^u), utilizados em Γ processos de autenticação válidos. Considerando que a chave x contém k bits, $k \geq m$, esta pode ser representada por $x = [x_{k-1}, x_{k-2}, ..., x_{m-1}, x_{m-2}, ..., x_2, x_1, x_0]$. Após ocorrer o primeiro deslocamento circular de x, i.e., $x = [x_{k-2}, x_{k-3}, ..., x_{m-2}, x_{m-3}, ..., x_1, x_0, x_{k-1}]$, os m bits utilizados na primeira etapa deste protocolo são $x_m = [x_{m-1}, x_{m-2}, ..., x_2, x_1, x_0]$ ou $x_m = [x_{m-2}, x_{m-3}, ..., x_2, x_1, x_0, x_{k-1}]$. Desta forma, o primeiro deslocamento, se ocorrer, em função do primeiro bit da chave y, o bit y_0, é desconsiderado do ponto de vista de criptoanálise, porque não faz diferença para o atacante determinar x_0 ou x_{k-1}, e sim o bit efetivamente utilizado, i.e., o conteúdo da posição zero. Para cada autenticação, os m bits menos significativos de x, os x_m bits, na j-ésima etapa, são formados pela mesma sequência de bits.

Nesta proposta de ataque consideramos que o atacante não tem controle na geração do bit aleatório ν de ruído, com probabilidade η de ser igual a 1. Para o atacante determinar o i-ésimo bit da chave x, o bit x_i, deve realizar alteração de um bit no desafio gerado pela leitora, para cada nova tentativa de autenticação. Para tanto, o atacante calcula $c^0 = a_0^0 \otimes b_0^0$ a partir do par de desafios (a_0^0, b_0^0), gerado na primeira etapa do primeiro processo de autenticação válido, e armazenado pelo atacante. Todo bit $c_i^0 = 1$, para $i = 0, m-1$, deve ser analisado.

Quando $c_i^0 = 1$ e $a_{0i}^0 = 1$, este bit deve ser alterado para zero, $a_{0i}^{'0} = 0$, resultando a sequência $a_0^{'0}$. Em um novo processo de autenticação, o atacante deve enviar para a etiqueta como desafio a sequência $a_0^{'0}$, alterada em apenas um bit em relação à sequência original a_0^0. Como a etiqueta gera bits aleatórios ν e também gera sequências b de desafios até que $a_0^{'0} \odot x_m \oplus v = b \odot x_m$, é necessário que o atacante interfira no processo de geração dessas sequências b. Seguindo as etapas do protocolo HB-MP, e baseado no nosso método de ataque, dada uma sequência $a_0^{'0}$, a primeira sequência b candidata deve ser aquela originalmente utilizada pela etiqueta, b_0^0. Com isto, a etiqueta deverá retornar para a leitora esta sequência ou outra qualquer. Se a etiqueta retornar como desafio o próprio b_0^0, o atacante deve adicionar a um contador este resultado como uma resposta válida. Este

processo deve ser repetido por 10 vezes. Se ao final desta parte do ataque, o resultado do contador for próximo a $10 \times (1 - \eta)$ é porque o bit x_i é igual a zero ($x_i = 0$), caso contrário $x_i = 1$. Este método de ataque deve ser executado para cada $c_i^0 = 1$ e $a_{0i}^0 = 1$, separadamente, determinando-se assim os bits da chave x correspondentes a estes bits c_i^0.

Em seguida, o atacante deve utilizar-se do par (a_0^1, b_0^1) e calcular $c^1 = a_0^1 \otimes b_0^1$. Para cada bit x_i da chave ainda não determinado, se $c_i^1 = 1$ e $a_{0i}^1 = 1$, o atacante deve realizar esta análise de forma equivalente à anterior, quando se analisou cada bit c_i^0, da sequência c^0. Análises sucessivas das etapas seguintes de cada autenticação devem ocorrer para cada bit c_i^u até que todos os m bits da chave x tenham sido determinados. Esta parte do ataque termina após a análise de Γ pares de desafios, em média, (a_0^u, b_0^u), para $u = 0, \Gamma - 1$. Para as etapas seguintes, utilizando-se dos pares (a_j^u, b_j^u), para $u = 2, \Gamma - 1$ e $j = 0, m - 1$, determinam-se os $r - 1$ bits restantes da chave x, porque esta sofre até $r - 1$ deslocamentos circular em cada etapa..

Este nosso Método de Ataque II pode ser representado em pseudo-código segundo o algoritmo 9:

Algorithm 9 Método de Ataque II ao Protocolo HB-MP.

Require: Coleção de Γ pares de desafios (a_j^u, b_j^u), $j = 0, r-1$, e $u = 0, \Gamma - 1$, segundo Tabela 7.1.

Ensure: Determinação dos $m+r-1$ bits da chave x, e dos $r-1$ bits da chaves y. $\chi^{r \times m}$ armazena os m bits de cada chave x utilizada em cada um dos r passos do protocolo.

for $j = 0$ to $r-1$ **do**
  $w \leftarrow 0$
  $s \leftarrow 0$
  for $u = 0$ to $\Gamma - 1$ **do**
    $c^u = a_j^u \otimes b_j^u$ ($c_i^u : i - esimo\ bit\ de\ c^u$)
    for $i = 0$ to $m-1$ **do**
      if $c_i^u = 1$ e $a_{ji}^u = 1$ **then**
        $a_j^{'u} = a_j^u$
        $a_{ji}^{'u} = 0$
        for $t = 1$ to 10 **do**
          a leitora envia para a etiqueta $a_j^{'u}$
          a etiqueta faz $b_j^{'u} \leftarrow b_j^u$
          a etiqueta testa $a_j^{'u}$ *com* $b_j^{'u}$
          while $a_j^{'u} \odot x_m \oplus v \neq b_j^{'u} \odot x_m$ **do**
            $gera - se\ outro\ b_j^{'u}\ compativel$
          end while
          A etiqueta retorna $b_j^{'u}$
          if $b_j^{'u} = b_j^u$ **then**
            $s \leftarrow s + 1$
          end if
        end for
        if $s \approx 10 \times (1 - \eta)$ **then**
          $\chi_i^j = x_i = 0$
        else
          $\chi_i^j = x_i = 1$
        end if
        $w \leftarrow w + 1$
      end if
    end for
  end for
  if $w = m$ **then**
    FIM
  end if
end for
return χ

7.6.2 Última Etapa $r+1$ do Método de Ataque II

Nesta última parte deste ataque, etapa $r+1$, faltam determinar os $r-1$ bits da chave y. As análises devem ocorrer em função dos resultados obtidos em relação à chave x, em cada uma das $r-1$ últimas etapas deste protocolo, segundo o algoritmo 9. Do ponto de vista desta análise, é irrelevante a determinação de y_0 porque os m bits menos significativos da chave x a serem considerados serão $x_m = [x_{m-2}, x_{m-3}, ..., x_2, x_1, x_0, x_{k-1}]$ ou $x_m = [x_{k-1}, x_{m-2}, x_{m-3}, ..., x_2, x_1, x_0]$, caso $y_0 = 1$ ou $y_0 = 0$, respectivamente. Para o atacante, o objetivo é a determinação da chave x efetivamente utilizada. Nesta etapa, a análise corresponde apenas em observar se a chave x foi deslocada entre a etapa j e a $j+1$, se o deslocamento ocorreu implica que o j-ésimo bit da chave y é igual a um, i.e., $y_j = 1$; caso contrário, $y_j = 0$.

7.7 Conclusões

Os resultados obtidos pelos métodos de ataque que criamos comprovam que sob as condições pré-estabelecidas, como as apresentadas para os métodos de ataque I e II, que a segurança do protocolo HB-MP pode ser bastante reduzida. O método I de ataque corresponde a realização de exaustão em apenas uma das chaves e a determinação da outra; o método II requer que o atacante tenha algum controle sobre a etiqueta durante o processo de geração do primeiro desafio para cada autenticação. Utilizando-se de um ou de outro método, o atacante determina todos os bits efetivamente utilizados das duas chaves, implementadas e compartilhadas entre a leitora e a etiqueta, para este protocolo de autenticação.

Capítulo 8

Ataque por Falha a Algoritmos Sequenciais Baseados em Autômato Celular

8.1 Introdução

Neste capítulo nós apresentamos um método de ataque por falha a um algoritmos criptográfico sequencial baseados em Autômato Celular, regra 30, proposto por Wolfram [66] e [67]. Esta classe de algoritmos criptográficos é empregada em processos de cifração e de decifração da informação com o objetivo de mantê-la em sigilo a terceiros não autorizados, a partir de uma uma chave simétrica. Assumimos que o atacante não tem conhecimento prévio da chave, mas pode trocar, provocar falha, em qualquer um dos seus bits (se for zero torna-se um, e vice-versa). O atacante pode também sempre que necessário reinicializar o dispositivo criptográfico e aplicar novas falhas com o objetivo de extrair informação suficiente para determinação de cada bit da chave. Para o Autômato Celular, regra 30, Wolfram [67] e [66], nós criamos um ataque prático por falha, onde é possível determinar todos os bits da chave com comprimento N qualquer, com apenas $\frac{N}{2} + \frac{N^2}{32}$ falhas, em média.

8.2 Pressupostos para o Ataque por Falha a Autômato Celular, regra 30

O algoritmo criptográfico sequencial em questão tem boas características aleatórias, Wolfram [67], o que é um quesito mínimo para um algoritmo criptográfico. Descrevemos brevemente a cifra sequencial em questão. O algoritmo consiste de um registrador circular com N células, cada uma tendo o valor a_j^t igual a zero ou a um, onde t representa o instante em que o conteúdo da célula é gerado, e i é a i-ésima célula do registrador. Estes valores são atualizados de forma sincronizada com base na seguinte regra:

$$a_i^t = a_{i-1}^{t-1}.xor.(a_i^{t-1}.or.a_{i+1}^{t-1}) \tag{8.1}$$

Ao se utilizar deste Autômato Celular como base de um algoritmo sequencial para cifração e decifração da informação, os bits $a_{N/2+1}^t$ são extraídos a cada instante da célula central do registrador, com comprimento N, e somados módulo dois à informação.

Assume-se que o atacante não tem conhecimento da chave, mas pode provocar falha em qualquer um dos seus bits (se for zero torna-se um, e vice-versa) armazenados na memória RAM ou no registrador interno ao dispositivo criptográfico onde estiver implementado este algoritmo. O atacante pode também, sempre que necessário, reinicializar o dispositivo criptográfico e aplicar uma falha, uma troca de bits, no registrador associado à chave criptográfica.

O nosso ataque consiste em o atacante coletar a sequência $a_{N/2+1}^t$, para $t = 1, ..., N/2 + 1$, e armazená-la na coluna central de uma matriz A, com dimensão $(N/2 + 1) \times N$. O elemento $a_i^t \in A$ está na i-ésima linha e t-ésima coluna de A. O método de ataque para determinação da chave se divide basicamente em: (i) à direita da coluna central da matriz A, montar uma matriz triangular superior à diagonal secundária; (ii) à esquerda da coluna central da matriz A, montar uma matriz triangular superior à diagonal principal. A chave estará determinada após o atacante provocar $\frac{N}{2} + \frac{N^2}{32}$ falhas no registrador e realizar análises dos resultados anteriores e posteriores à essas falhas. Neste ataque, a quantidade de bits necessária para se quebrar a chave corresponde apenas à metade do tamanho do registrador.

8.3 Método de Ataque por Falha a Algoritmos Sequenciais Baseados em Autômato Celular, regra 30

Para se aplicar o método de ataque que nós criamos para algoritmos baseados em Autômato Celular, regra 30, o atacante deve armazenar na coluna central de uma matriz A, com dimensão $(N/2 + 1) \times N$, a sequência de bits $a_{N/2+1}^t$, para $t = 1, ..., N/2 + 1$. Esta sequência é extraída da célula central do registrador com N bits, em $(N/2 + 1)$ etapas. O elemento $a_i^t \in A$ está na i-ésima linha e na t-ésima coluna. Este ataque pressupõe o conhecimento de apenas $N/2 + 1$ bits por parte do atacante para se determinar toda a chave secreta. Esta sequência de bits pode ser extraída de qualquer posição da sequência cifrada, sem qualquer restrição, com apenas a metade do comprimento do registrador.

A aplicação do nosso método de ataque é simples, compreende a análise de pares de bits da matriz A, contidos em suas colunas. Este método é dividido em três etapas: a primeira etapa consiste na determinação dos bits das duas colunas vizinhas à coluna central da matriz A, à esquerda e à direita; na segunda etapa determinam-se os bits correspondentes às colunas à direita da coluna central, formando uma matriz triangular superior à diagonal secundária; e na terceira etapa, por último, determinam-se os bits das colunas à esquerda da coluna central, formando uma matriz triangular superior à diagonal principal. O método é bastante prático e eficiente, tem como resultado a extração de toda a chave após provocar apenas $\frac{N}{2} + \left(\frac{\left(\frac{N}{2}\right) \times \left(\frac{N}{2}\right)}{2} \right) \div 4 = \frac{N}{2} + \frac{N^2}{32}$ falhas em média no registrador. Cada bit da chave é determinado a partir dos bits armazenados na coluna

central da matriz A, adicionados àqueles que são determinados por este ataque e são armazenados à direita e à esquerda desta coluna, formando duas matrizes triangulares superiores. O método de ataque corresponde ao atacante provocar falhas e realizar análises nos bits contidos nessas colunas das matrizes triangulares. Para se determinar a matriz triangular à direita da coluna central, a análise deve ser realizada coluna por coluna da esquerda para a direita. Para se determinar a matriz triangular à esquerda da coluna central, a análise deve ser realizada coluna por coluna da direita para a esquerda. Como exemplo, se a chave secreta tiver 256 bits, aplicando-se este método de ataque, é necessário realizar apenas $2^7 + 2^{11} = 2176$ falhas, acrescido de análises comparativas, para se determinar todos os bits desta chave.

Considerando a equação 8.1, que atualiza os bits das células do registrador, o atacante deve iniciar este método de ataque pelos dois últimos bits da sequência conhecida, que foi armazenada na coluna central da matriz A, em suas duas últimas linhas: $a_{N/2+1}^{N/2+1}$ e $a_{N/2+1}^{N/2}$. Observando-se este par de bits e aplicando-se uma falha em um dos dois bits, os bits $a_{N/2}^{N/2}$ e $a_{N/2+2}^{N/2}$ são determinados. Em seguida os bits $a_{N/2+1}^{N/2}$ e $a_{N/2+1}^{N/2-1}$ devem ser analisados, determinando-se assim os bits $a_{N/2}^{N/2-1}$ e $a_{N/2+2}^{N/2-1}$. O atacante ao analisar os bits da coluna central de dois a dois determinará os seus vizinhos. Esta análise deverá ocorrer até que se analise os dois primeiros bits desta sequência, que são $a_{N/2+1}^{1}$ e $a_{N/2+1}^{0}$ e determinar os seus bits vizinhos $a_{N/2}^{0}$ e $a_{N/2+2}^{0}$.

Uma vez determinados os bits das duas colunas vizinhas à coluna central, o atacante deve analisar os bits da coluna à direita da coluna central, os bits $a_{N/2+2}^{t}$ que foram determinados na análise anterior. Neste momento se conhecem três dos quatro bits envolvidos na análise, i.e., os bits $a_{N/2+1}^{t}$, $a_{N/2+2}^{t}$ e $a_{N/2+2}^{t+1}$, faltando-se determinar apenas os bits da coluna $a_{N/2+3}^{t}$. Ao se determinar os bits desta coluna, deve-se passar para a próxima coluna à direita, e assim sucessivamente, mantendo-se esta metodologia de ataque até se determinar o bit a_{N}^{0}, que é o último bit da chave à direita da coluna central. Nesta etapa aplicam-se falhas em apenas $\frac{1}{4}$ dos pares de bits analisados, em média, como será descrito em detalhe, posteriormente.

Ao se terminar a análise dos pares de bits que estão à direita da coluna central, a metade dos bits da chave secreta está determinada. Então, falta a outra metade dos bits, àqueles que estão à esquerda desta coluna.

A análise prossegue de forma análoga, nesta etapa sem necessidade de se aplicar qualquer falha nos pares de bits analisados. O ataque consiste em o atacante analisar cada coluna à esquerda da coluna central, uma a uma, da direita para a esquerda. A primeira coluna a ser analisada corresponde aos bits $a_{N/2}^{t}$ que foram determinados na primeira etapa este ataque. Para esta parte da análise, se conhecem três dos quatro bits envolvidos, i.e., os bits $a_{N/2+1}^{t}$, $a_{N/2}^{t}$ e $a_{N/2}^{t+1}$, faltando-se determinar os bits da coluna anterior, $a_{N/2-1}^{t}$. Ao se determinar os bits desta coluna a partir da relação algébrica correspondente a equação 8.1 deve-se passar para a próxima coluna à esquerda, e assim sucessivamente, mantendo o método de ataque até se determinar o bit a_{0}^{0}. Nesse momento, todos os bits da chave secreta foram determinados.

Etapa 1: **Determinação dos bits das Colunas Vizinhas à Coluna Central**

Nesta etapa, para cada par de bits conhecidos, aplica-se uma falha no registrador. Com uma simples análise do resultado é possível determinar os dois pares de bits vizinhos à coluna analisada. Como são pares de bits, existem apenas quatro casos que devemos analisar: $\{\begin{pmatrix} 0 \\ 0 \end{pmatrix}, \begin{pmatrix} 0 \\ 1 \end{pmatrix}, \begin{pmatrix} 1 \\ 0 \end{pmatrix}, \begin{pmatrix} 1 \\ 1 \end{pmatrix}\}$. Aqui representaremos estes pares na seguinte ordem $\begin{pmatrix} \alpha \\ \beta \end{pmatrix} = \begin{pmatrix} a_i^{t-1} \\ a_i^t \end{pmatrix}$. Baseado na equação 8.1 e de posse dos bits a_i^{t-1} e a_i^t é possível, aplicando uma única falha no registrador, determinar os bits a_{i-1}^{t-1} e a_{i+1}^{t-1}, i.e., $\begin{pmatrix} a_{i-1}^{t-1} & a_i^{t-1} & a_{i+1}^{t-1} \\ x & a_i^t & x \end{pmatrix}$.

Analisaremos em seguida cada um dos quatro pares de bits e mostraremos como determinar os dois bits vizinhos:

1. $\begin{pmatrix} a_{i-1}^{t-1} & a_i^{t-1} & a_{i+1}^{t-1} \\ x & a_i^t & x \end{pmatrix} = \begin{pmatrix} a_{i-1}^{t-1} & 0 & a_{i+1}^{t-1} \\ x & 0 & x \end{pmatrix}$:

   Esta condição só é possível se $a_{i-1}^{t-1} = a_{i+1}^{t-1} = 1$ ou se $a_{i-1}^{t-1} = a_{i+1}^{t-1} = 0$. Se provocarmos uma falha no bit $a_i^{t-1} = 0$, trocá-lo para um, e recalcular o bit a_i^t, só existem duas possibilidades:

   - se $a_i^t = 0$, então $a_{i-1}^{t-1} = a_{i+1}^{t-1} = 1$
   - se $a_i^t = 1$, então $a_{i-1}^{t-1} = a_{i+1}^{t-1} = 0$

2. $\begin{pmatrix} a_{i-1}^{t-1} & a_i^{t-1} & a_{i+1}^{t-1} \\ x & a_i^t & x \end{pmatrix} = \begin{pmatrix} a_{i-1}^{t-1} & 0 & a_{i+1}^{t-1} \\ x & 1 & x \end{pmatrix}$:

   Esta condição só é possível se $a_{i-1}^{t-1} = 0$ e $a_{i+1}^{t-1} = 1$ ou se $a_{i-1}^{t-1} = 1$ e $a_{i+1}^{t-1} = 0$. Se provocarmos uma falha no bit $a_i^{t-1} = 0$, trocá-lo para um, e recalcular o bit $a_i^t = 0$, só existem duas possibilidades:

   - se $a_i^t = 1$, então $a_{i-1}^{t-1} = 0$ e $a_{i+1}^{t-1} = 1$
   - se $a_i^t = 0$, então $a_{i-1}^{t-1} = 1$ e $a_{i+1}^{t-1} = 0$

3. $\begin{pmatrix} a_{i-1}^{t-1} & a_i^{t-1} & a_{i+1}^{t-1} \\ x & a_i^t & x \end{pmatrix} = \begin{pmatrix} a_{i-1}^{t-1} & 1 & a_{i+1}^{t-1} \\ x & 0 & x \end{pmatrix}$

   Esta condição define de imediato $a_{i-1}^{t-1} = 1$, e mantém indefinido a_{i+1}^{t-1}. Se provocarmos uma falha no bit $a_i^{t-1} = 1$, trocá-lo para zero, e recalcular o bit a_i^t, só existem duas possibilidades:

   - se $a_i^t = 1$, então $a_{i-1}^{t-1} = 1$ and $a_{i+1}^{t-1} = 0$
   - se $a_i^t = 0$, então $a_{i-1}^{t-1} = 1$ and $a_{i+1}^{t-1} = 1$

4. $\begin{pmatrix} a_{i-1}^{t-1} & a_i^{t-1} & a_{i+1}^{t-1} \\ x & a_i^t & x \end{pmatrix} = \begin{pmatrix} a_{i-1}^{t-1} & 1 & a_{i+1}^{t-1} \\ x & 1 & x \end{pmatrix}$:

   Esta condição define de imediato $a_{i-1}^{t-1} = 0$, e mantém indefinido a_{i+1}^{t-1}. Se provocarmos uma falha no bit $a_i^{t-1} = 1$, trocá-lo para zero, e recalcular o bit a_i^t, só existem duas possibilidades:

   - se $a_i^t = 1$, então $a_{i-1}^{t-1} = 0$ and $a_{i+1}^{t-1} = 1$
   - se $a_i^t = 0$, então $a_{i-1}^{t-1} = 0$ and $a_{i+1}^{t-1} = 0$

Etapa 2: **Determinação dos bits das Colunas à Direita da Coluna Central**

Nesta etapa, em apenas $\frac{1}{4}$ dos ternos de bits conhecidos, em média, aplica-se falha no registrador. Como resultado é possível definir o bit vizinho à direita da coluna analisada, ainda desconhecido. Para a metade, em média, dos ternos de bits basta realizar análises comparativas simples para determinação de cada outro bit à direita, enquanto que os outros $\frac{1}{4}$ possíveis não ocorrem. Como são ternos de bits, existem apenas oito casos que devemos analisar: $\{\begin{pmatrix} 0 & 0 \\ x & 0 \end{pmatrix}, \begin{pmatrix} 0 & 0 \\ x & 1 \end{pmatrix}, \begin{pmatrix} 0 & 1 \\ x & 0 \end{pmatrix}, \begin{pmatrix} 0 & 1 \\ x & 1 \end{pmatrix}\}$ ou $\{\begin{pmatrix} 1 & 0 \\ x & 0 \end{pmatrix}, \begin{pmatrix} 1 & 0 \\ x & 1 \end{pmatrix}, \begin{pmatrix} 1 & 1 \\ x & 0 \end{pmatrix}, \begin{pmatrix} 1 & 1 \\ x & 1 \end{pmatrix}\}$. Aqui representaremos estes ternos na seguinte ordem $\begin{pmatrix} \alpha & \beta \\ x & \delta \end{pmatrix} = \begin{pmatrix} a_{i-1}^{t-1} & a_i^{t-1} \\ x & a_i^t \end{pmatrix}$. Baseado na equação 8.1 e de posse dos bits a_{i-1}^{t-1}, a_i^{t-1} e a_i^t é possível, aplicando uma única falha no registrador, determinar o bit a_{i+1}^{t-1}, i.e., $\begin{pmatrix} a_{i-1}^{t-1} & a_i^{t-1} & a_{i+1}^{t-1} \\ x & a_i^t & x \end{pmatrix}$.

Analisaremos em seguida cada um desses ternos de bits e mostraremos como determinar o quarto bit, à direita da coluna analisada:

1. $\begin{pmatrix} a_{i-1}^{t-1} & a_i^{t-1} & a_{i+1}^{t-1} \\ x & a_i^t & x \end{pmatrix} = \begin{pmatrix} 0 & 0 & a_{i+1}^{t-1} \\ x & 0 & x \end{pmatrix}$:

   Esta condição só é possível para $a_{i+1}^{t-1} = 0$

2. $\begin{pmatrix} a_{i-1}^{t-1} & a_i^{t-1} & a_{i+1}^{t-1} \\ x & a_i^t & x \end{pmatrix} = \begin{pmatrix} 0 & 0 & a_{i+1}^{t-1} \\ x & 1 & x \end{pmatrix}$:

   Esta condição só é possível para $a_{i+1}^{t-1} = 1$

3. $\begin{pmatrix} a_{i-1}^{t-1} & a_i^{t-1} & a_{i+1}^{t-1} \\ x & a_i^t & x \end{pmatrix} = \begin{pmatrix} 0 & 1 & a_{i+1}^{t-1} \\ x & 0 & x \end{pmatrix}$:

   Esta condição não é possível, não ocorre.

4. $\begin{pmatrix} a_{i-1}^{t-1} & a_i^{t-1} & a_{i+1}^{t-1} \\ x & a_i^t & x \end{pmatrix} = \begin{pmatrix} 1 & 1 & a_{i+1}^{t-1} \\ x & 1 & x \end{pmatrix}$:

   Se provocarmos uma falha no bit $a_i^{t-1} = 1$, trocá-lo para zero, e recalcular o bit a_i^t, só existem duas possibilidades:

   (a) Se o resultado for $a_i^t = 1$, então $a_{i+1}^{t-1} = 1$
   (b) Se o resultado for $a_i^t = 0$, então $a_{i+1}^{t-1} = 0$

5. $\begin{pmatrix} a_{i-1}^{t-1} & a_i^{t-1} & a_{i+1}^{t-1} \\ x & a_i^t & x \end{pmatrix} = \begin{pmatrix} 1 & 0 & a_{i+1}^{t-1} \\ x & 0 & x \end{pmatrix}$:

   Esta condição só é possível para $a_{i+1}^{t-1} = 1$

6. $\begin{pmatrix} a_{i-1}^{t-1} & a_i^{t-1} & a_{i+1}^{t-1} \\ x & a_i^t & x \end{pmatrix} = \begin{pmatrix} 1 & 0 & a_{i+1}^{t-1} \\ x & 1 & x \end{pmatrix}$:

   Esta condição só é possível para $a_{i+1}^{t-1} = 0$

7. $\begin{pmatrix} a_{i-1}^{t-1} & a_i^{t-1} & a_{i+1}^{t-1} \\ x & a_i^t & x \end{pmatrix} = \begin{pmatrix} 1 & 1 & a_{i+1}^{t-1} \\ x & 0 & x \end{pmatrix}$

   Se provocarmos uma falha no bit $a_i^{t-1} = 1$, trocá-lo para zero, e recalcular o bit a_i^t, só existem duas possibilidades:

   (a) Se o resultado for $a_i^t = 1$, então $a_{i+1}^{t-1} = 0$
   (b) Se o resultado for $a_i^t = 0$, então $a_{i+1}^{t-1} = 1$

8. $\begin{pmatrix} a_{i-1}^{t-1} & a_i^{t-1} & a_{i+1}^{t-1} \\ x & a_i^t & x \end{pmatrix} = \begin{pmatrix} 1 & 1 & a_{i+1}^{t-1} \\ x & 1 & x \end{pmatrix}$

   Esta condição não é possível, não ocorre.

Etapa 3: **Determinação dos bits das Colunas à Esquerda da Coluna Central**

Nesta etapa, para cada terno de bits conhecidos é possível determinar o bit vizinho à esquerda da coluna analisada, sem necessidade de se aplicar qualquer falha no registrador. Apenas com análises comparativas simples, todos os bits da matriz triangular superior à diagonal principal, à esquerda da coluna central da matriz A, são determinados. Como são ternos de bits a serem

analisados, existem apenas oito casos que devemos considerar: $\{ \begin{pmatrix} 0 & 0 \\ 0 & x \end{pmatrix}, \begin{pmatrix} 0 & 0 \\ 1 & x \end{pmatrix}, \begin{pmatrix} 1 & 0 \\ 0 & x \end{pmatrix}, \begin{pmatrix} 1 & 0 \\ 1 & x \end{pmatrix}, \begin{pmatrix} 0 & 1 \\ 0 & x \end{pmatrix}, \begin{pmatrix} 0 & 1 \\ 1 & x \end{pmatrix}, \begin{pmatrix} 1 & 1 \\ 0 & x \end{pmatrix}, \begin{pmatrix} 1 & 1 \\ 1 & x \end{pmatrix}\}$. Aqui representaremos estes ternos na seguinte ordem $\begin{pmatrix} \alpha & \beta \\ \delta & x \end{pmatrix} = \begin{pmatrix} a_{i-1}^{t-1} & a_i^{t-1} \\ a_{i-1}^t & x \end{pmatrix}$. Baseado na equação 8.1 e de posse dos bits a_{i-1}^t, a_{i-1}^{t-1} e a_i^{t-1} é possível determinar o bit a_{i-2}^{t-1}, i.e., $\begin{pmatrix} a_{i-2}^{t-1} & a_{i-1}^{t-1} & a_i^{t-1} \\ x & a_{i-1}^t & x \end{pmatrix}$.

Analisaremos em seguida cada um desses ternos de bits e mostraremos como determinar o quarto bit, à esquerda da coluna analisada:

1. $\begin{pmatrix} a_{i-2}^{t-1} & a_{i-1}^{t-1} & a_i^{t-1} \\ x & a_{i-1}^t & x \end{pmatrix} = \begin{pmatrix} a_{i-2}^{t-1} & 0 & 0 \\ x & 0 & x \end{pmatrix}$:

   Esta condição só é possível para $a_{i-2}^{t-1} = 0$

2. $\begin{pmatrix} a_{i-2}^{t-1} & a_{i-1}^{t-1} & a_i^{t-1} \\ x & a_{i-1}^t & x \end{pmatrix} = \begin{pmatrix} a_{i-2}^{t-1} & 0 & 0 \\ x & 1 & x \end{pmatrix}$:

   Esta condição só é possível para $a_{i-2}^{t-1} = 1$

3. $\begin{pmatrix} a_{i-2}^{t-1} & a_{i-1}^{t-1} & a_i^{t-1} \\ x & a_{i-1}^t & x \end{pmatrix} = \begin{pmatrix} a_{i-2}^{t-1} & 1 & 0 \\ x & 0 & x \end{pmatrix}$:

   Esta condição só é possível para $a_{i-2}^{t-1} = 1$

4. $\begin{pmatrix} a_{i-2}^{t-1} & a_{i-1}^{t-1} & a_i^{t-1} \\ x & a_{i-1}^t & x \end{pmatrix} = \begin{pmatrix} a_{i-2}^{t-1} & 1 & 0 \\ x & 1 & x \end{pmatrix}$:

   Esta condição só é possível para $a_{i-2}^{t-1} = 0$

5. $\begin{pmatrix} a_{i-2}^{t-1} & a_{i-1}^{t-1} & a_i^{t-1} \\ x & a_{i-1}^t & x \end{pmatrix} = \begin{pmatrix} a_{i-2}^{t-1} & 0 & 1 \\ x & 0 & x \end{pmatrix}$:

   Esta condição só é possível para $a_{i-2}^{t-1} = 1$

6. $\begin{pmatrix} a_{i-2}^{t-1} & a_{i-1}^{t-1} & a_i^{t-1} \\ x & a_{i-1}^t & x \end{pmatrix} = \begin{pmatrix} a_{i-2}^{t-1} & 0 & 1 \\ x & 1 & x \end{pmatrix}$

   Esta condição só é possível para $a_{i-2}^{t-1} = 0$

7. $\begin{pmatrix} a_{i-2}^{t-1} & a_{i-1}^{t-1} & a_i^{t-1} \\ x & a_{i-1}^{t} & x \end{pmatrix} = \begin{pmatrix} a_{i-2}^{t-1} & 1 & 1 \\ x & 0 & x \end{pmatrix}$:

   Esta condição só é possível para $a_{i-2}^{t-1} = 1$

8. $\begin{pmatrix} a_{i-2}^{t-1} & a_{i-1}^{t-1} & a_i^{t-1} \\ x & a_{i-1}^{t} & x \end{pmatrix} = \begin{pmatrix} a_{i-2}^{t-1} & 1 & 1 \\ x & 1 & x \end{pmatrix}$:

   Esta condição só é possível para $a_{i-2}^{t-1} = 0$

8.4 Conclusões

Os melhores resultados anteriormente publicados em ataques a algoritmos baseados em Autômato Celular foram obtidos pelo método proposto por Meier e Staffelbach [48]. Com este método de ataque consegue-se determinar chaves com tamanhos entre 300 e 500 bits (dependendo da regra de atualização dos bits no registrador), pressupondo o conhecimento do claro, e utilizando-se de estimativas estatísticas e de computadores pessoais. Os autores também reforçam a possibilidade de determinação de chaves com comprimentos maiores, com aproximadamente 1000, utilizando-se de computadores de grande porte com *hardware* especial e com processamento paralelo.

Algoritmos baseados em Autômato Celular, com quaisquer tamanhos de chaves, não resistem ao nosso método de ataque por falha. A complexidade computacional do método que propusemos para determinação da chave é desprezível, bastante inferior aos resultados já publicados. Utilizando-se do nosso método de ataque, para uma chave com 256 bits, por exemplo, é necessário o conhecimento de apenas $256/2+1 = 129$ bits da sequência gerada, e de se provocar apenas $\frac{N}{2} + \frac{N^2}{32} = \frac{256}{2} + \frac{256^2}{32} = 2^7 + 2^{11}$ falhas no registrador do dispositivo, determinando-se todos os bits da chave a partir de análises simples, sendo em boa parte apenas análises comparativas. Quando a chave tiver 1024 bits, por exemplo, são necessários apenas $1024/2 + 1 = 513$ bits conhecidos da sequência gerada, determinando-se toda a chave com apenas $\frac{N}{2} + \frac{N^2}{32} = \frac{1024}{2} + \frac{1024^2}{32} = 2^9 + 2^{15}$ falhas no registrador. O nosso método de ataque é bastante robusto, rápido e eficiente para qualquer tamanho de chave.

Capítulo 9

Conclusões

O objetivo principal do nosso trabalho de pequisa foi o de criar novos e eficientes métodos de ataques criptoanalíticos a protocolos direcionados a dispositivos com baixo poder de processamento, como os RFID. Nós criamos diversos métodos distintos de ataques para analisar protocolos da família de protocolos HB, como o próprio HB, o HB^{+}, o HB^{++} (três versões), o HB*, o HB$^{\#}$, o Random-HB$^{\#}$, e o HB-MP. Analisamos também protocolos baseados no problema da soma de k mínimos, e algoritmos sequenciais baseados em Autômato Celular, regra 30. Os nossos ataques são basicamente de três tipos distintos: (passivo) probabilístico, passivo, e por falha. O que diferencia principalmente o nosso trabalho daqueles já publicados é que a quase totalidade dos ataques já apresentados na literatura a esses protocolos são do tipo *man-in-the-middle*, enquanto que nós criamos uma variedade de diferentes métodos de ataques para cada um desses protocolos ou algoritmos.

O nosso método de ataque passivo e probabilístico aos protocolos HB e HB^{+} muda o enforque sobre os métodos de ataques já propostos na literatura para essa classe de protocolos. Nós verificamos, baseados nos resultados que obtivemos, que o ataque probabilístico é um método prático e bastante eficiente, sendo melhor que os métodos BKW e FMICM, para vários parâmetros, sendo esses atualmente a melhor referência técnica de ataque passivo publicada. O método proposto não requer pré-processamento, nem prévio conhecimento do percentual de ruído. A quantidade de pares desafios-resposta necessária para o nosso ataque é bastante pequena, principalmente se comparada a grande quantidade exigida por esses dois outros métodos, que por este motivo os tornam ataques teóricos, para a maioria dos parâmetros. Além disso, a aplicação do nosso método de ataque permite que o atacante faça uma relação de conveniência entre o esforço computacional e a quantidade de pares desafios-resposta necessária para quebrá-los: ora o atacante pode optar por aumentar o esforço computacional e reduzir a quantidade de pares de desafios-resposta, ora o atacante de posse de menos processamento aumenta quantidade de desafios-resposta, conforme conveniência. O mesmo não é possível para os métodos BKW e FMICM, onde, em qualquer caso, a quantidade de pares desafios-resposta cresce exponencialmente com o comprimento da chave. Portanto, o nosso método de escolha aleatória de subconjuntos de desafios, agregado a um método bem elaborado de análise e de decisão para determinação do resultado correto da chave, é um excelente método criptoanalítico para se quebrar esses protocolos, principalmente se levarmos em considerando que esses protocolos têm segurança comprovada baseada no *Learning Problem with*

noise - LPN.

Os resultados obtidos com a aplicação do método de ataque por falha que propusemos aos protocolos HB/HB$^+$ mostram-se bastante mais eficientes do que qualquer outro ataque já apresentado. É possível determinar a chave em muito poucas operações: para uma chave com 128 bits são necessários apenas 17 desafios, em média; para uma chave com 256 bits precisa-se de apenas 20 desafios; e para chaves com mais de 10000 bits são necessários apenas 31 desafios. Proporcionalmente, o aumento do número de desafios necessários para se determinar toda a chave cresce de forma bastante mais lenta com o comprimento da chave. O resultado obtido é prático e muito eficiente, com a sua aplicação torna-se possível a determinação de uma chave gravada em área de memória protegida em dispositivos criptográficos. Esse ataque independe inclusive do percentual de erro do ruído. Estes resultados evidenciam a fragilidade destes protocolos deixando-os vulneráveis para qualquer que seja a chave compartilhada.

O ataque ativo aqui proposto a protocolos baseados no problema da soma de k mínimos permite que o atacante determine toda a chave com um esforço computacional irrelevante. Para tanto, é suficiente que o atacante gere e escolha uma quantidade de sequências bem reduzida, calcule os desafios correspondentes, e determine os inteiros pertencentes à chave a partir de análises comparativas. Os nossos resultados são muito expressivos, mostram que um protocolo baseado no problema da soma de k mínimos não resiste ao nosso ataque, o seu uso pode deixar vulneráveis os dispositivos de autenticação e identificação onde estiverem implementados.

O nosso método de ataque passivo a protocolos baseados no problema da soma de k mínimos é simples e mais eficiente do que um processo exaustivo em todas as chaves possíveis. Esta nossa proposta também é mais eficiente do que um ataque semelhante ao de exaustão onde o atacante realiza uma pré-triagem de chaves e exaure as restantes.

Os resultados que obtivemos com os métodos de ataque ao protocolo HB-MP comprovam que sob algumas condições pré-estabelecidas este protocolo é bastante frágil. Nós atacamos o protocolo HB-MP utilizando-se de dois métodos de ataque que criamos: o primeiro método corresponde ao atacante exaurir uma das chaves e após alguma análise determinar a outra; o segundo método de ataque pressupõe que o atacante tem algum controle da etiqueta durante o processo de geração do primeiro desafio de alguns poucos processos de autenticação. Estes ataques comprovam que é possível determinar de forma muito eficiente todos os bits das chaves utilizadas neste protocolo de autenticação.

Nós criamos um novo ataque por falha a protocolos ou algoritmos sequenciais baseados em Autômato Celular, regra 30. Os resultados mostram-se bastante eficientes, indicando que o nosso ataque é robusto e rápido. A partir do nosso ataque podemos determinar a chave utilizada pelo algoritmo com quaisquer tamanhos, com muito poucas operações.

Os nossos resultados junto aos outros resultados já apresentados por diversos autores na literatura evidenciam a necessidade de se incentivar projetos nesta área, abrindo linhas de pesquisa tanto para criação como para análise de protocolos criptográficos destinados aos dispositivos de baixo poder computacional. Entendemos que o objetivo dessa linha de pesquisa a ser implementada deve ser o de diminuir a dependência de padrões criptográficos com processamento de suas operações muito pesado para uma arquitetura tão simplificada como a dos chips RFID.

REFERÊNCIAS BIBLIOGRÁFICAS

[1] Abadi, M., Burrows, M., and Kaufman, C. Authentication and delegation with smart-cards. *Theoretical Aspects of Computer Software*, 326–345.

[2] AlienTechnology. *web site http://www.alientechnology.com*.

[3] Anderson, R., and Kuhn, M. Low cost attacks on tamper resistant devices. *Security Protocols Workshop 1361* (1997), 125–136.

[4] Avoine, G. *IFIP World Computer Congress, Sixth Smart Card Research and Advanced Application IFIP Conference - CARDIS'04*.

[5] Avoine, G. Adversarial model for radio frequency identification. *Technical Report LASEC-REPORT-2005-001, Swiss Federal Institute of Technology (EPFL), Security and Cryptography Laboratory - LASEC - http://eprint.iacr.org/2005/049.pdf*, 1–14.

[6] Avoine, G., Dysli, E., and Oechslin, P. Reducing time complexity in rfid systems. *Selected Areas in Cryptography - SAC 2005 3897* (2005), 291–306.

[7] Bar-El, H., Choukri, H., Naccache, D., Tunstall, M., and Whelan, C. The sorcerer's apprenctice guide to fault attacks. *Technical report*, 1–13.

[8] Biehl, I., Meyer, B., and Müller, V. Differential fault attacks on elliptic curve cryptosystems. *Advances in Cryptology - CRYPTO 2000 1880* (2000), 131–146.

[9] Biham, E., and Shamir, A. Differential fault analysis: Identifying the structure of unknown ciphers sealed in tamper-proof devices. *preprint, 10/11/96*, 1–2.

[10] Biham, E., and Shamir, A. Differential fault analysis of secret key cryptosystems. *Advances in Cryptology - CRYPTO'97 1294* (1997), 513–525.

[11] Blum, A., Furst, M., Kearns, M., , and Lipton, R. Cryptographic primitives based on hard learning problems. *Advanced in Cryptology - Crypto 93 773* (1994), 278–291.

[12] Blum, A., Kalai, A., and Wasserman, H. Noise-tolerant learning, the parity problem, and the statistical query model. *Journal of the ACM 50(4)* (2003), 506–519.

[13] Boneh, D., DeMillo, R. A., and Lipton, R. J. On the importance of checking cryptographic protocols for faults. *Advanced in Cryptology - EUROCRYPT'97 1233* (1997), 37–51.

[14] Boneh, D., DeMillo, R. A., and Lipton, R. J. On the importance of eliminating errors in cryptographic computations. *Journal of Cryptology 14(2)* (2001), 101–119.

[15] Bringer, J., Chabanne, H., and Dottax, E. Hb++: A lightweight authentication protocol secure against some attacks. *IEEE Int'l Conf. Pervasive Service, Workshop Security, Privacy and Trust in Pervasive and Ubiquitous Computing, 2006*, 28–33.

[16] by CATAMARAN, E. C. P. P. *web site http://www.shipcomwireless.com/PDF/040604-Brochure-complaince.pdf*, 1–2.

[17] Carrijo, J., Tonicelliy, R., Imaiz, H., and Nascimento, A. C. A. A novel probabilistic passive attack on the protocols hb and hb+. *IEICE Transactions on Fundamentals of Electronics, Communications and Computer Science E92-A* (2009), Number 2, pp. 658–662.

[18] corporation, V. *web site http://www.verichipcorp.com*.

[19] Council, U. C. *web site web site http://www.uc-council.org*.

[20] Duc, D. N., and Kim, K. Securing hb+ against grs man-in-the-middle attack. *The 2007 Symposium on Cryptography and Information Security, SCIS 2007, IEICE, Sasebo, Japan*, 1–5.

[21] Fossorier, M., Mihaljevic, M., and Imai, H. A unified analysis for the fast correlation attack. *Proceedings of International Symposium on Information Theory - ISIT 4-9 Sept.* (2005), 2012 – 2015.

[22] Fossorier, M. P. C., Mihaljevi, M. J., Imai, H., Cuiz, Y., and Matsuura, K. A novel algorithm for solving the lpn problem and its application to security evaluation of the hb protocol for rfid authentication. *eprint.iacr.org/2006/197.pdf*, 1–14.

[23] Gamal, T. E. A public key cryptosystem and a signature scheme based on discrete logarithms. *IEEE Transactions on Information Theory. web site http://groups.csail.mit.edu/cis/crypto/classes/6.857/papers/elgamal.pdf 31* (1985), 469–472.

[24] Gilbert, H., Robshaw, M., and Sibert, H. An active attack against hb+ - a provably secure lightweight authentication protocol. *Cryptology ePrint Archive, Report 2005/237*, 237.

[25] Gilbert, H., Robshaw, M. J., and Seurin, Y. Hb$^\sharp$: Increasing the security and efficiency of hb$^+$. *Advances in Cryptology - EUROCRYPT 2008 4965* (2008), 361–378.

[26] Giraud, C., and Thiebeauld, H. A survey on fault attacks. *Proceedings of Smart Card Research and Advanced Applications VI - 18th IFIP World Computer Congress*, 159–176.

[27] Golebiewski, Z., Majcher, K., zagórski, F., and Zawada, M. Pratical attacks on hb and hb+ protocols. *ePrint Archive, web site http://eprint.iacr.org/2008/241.pdf*, 1–10.

[28] Good, N., Han, J., Miles, E., Molnar, D., Mulligan, D., Quilter, L., Urban, J. M., and Wagner, D. Radio frequency id and privacy with information goods. *Proceedings of ACM workshop on Privacy in the electronic society*, 41 – 42.

[29] GS1. *web site http://www.ean-int.org*.

[30] Hastad, J. Some optimal inapproximability results. *Journal of the ACM 48(4)* (2001), 798–859.

[31] Hoch, J. J., and Shamir, A. Fault analysis of stream ciphers. *Cryptographic Hardware and Embedded Systems - CHES 2004 3156* (2004), 240–253.

[32] Hopper, N., and Blum, M. A secure human-computer authentication scheme. *Technical Report CMU-CS-00-139, Carnegie Mellon University*, 1–6.

[33] Hopper, N. J., and Blum, M. Secure human identification protocols. *proceedings of ASIACRYPT 2001 2248* (2001), 52–66.

[34] Jr., B. S. K., and Robshaw, M. J. B. Comments on some new attacks on cryptographic devices. *RSA Laboratories' Bulletin No. 5, July 1997. ftp://ftp.rsa.com/pub/pdfs/bulletn5.pdf*, 1–5.

[35] Juels, A. Rfid security and privacy - a research survey. *RSA Laboratories*, 1–19.

[36] Juels, A. Strengthing epc tags against cloning. *Proceedings of the 4th ACM workshop on Wireless security*, 67 – 76.

[37] Juels, A., Molnar, D., and Wagner, D. Security and privacy issues in e-passports. *IEEE/CreateNet SecureComm 2005. web site http://www.cs.berkeley.edu/ dmolnar/papers/papers.htm*, 1–12.

[38] Juels, A., and Pappu, R. Squealing euros: Privacy protection in rfid-enabled banknotes. *proceedings of Financial Cryptography - FC'03 2742* (2003), 103–121.

[39] Juels, A., and Weis, S. A. Authenticating pervasive devices with human protocols. *proceedings of CRYPTO 2005 3621* (2005), 293–308.

[40] Katz, J., and Shin, J. S. Parallel and concurrent security of the hb and hb+ protocols. *Cryptology ePrint archive, Report 2005/461*, 1–13.

[41] Katz, J., and Smith, A. Analyzing the hb and hb+ protocols in the large error case. *Cryptology eprint archieve. web site eprint.iacr.org/2006/326.pdf*, 1–9.

[42] Kearns, M. Efficient noise-tolerant learning from statistical queries. *Journal of the ACM 45 , Issue 6 (November 1998)* (1998), 983–1006.

[43] Kfir, Z., and Wool, A. Picking virtual pockets using relay attacks on contactless smartcard systems. *In IEEE/CreateNet SecureComm. IEEE, 2005. web site ttp://eprint.iacr.org/2005/052*, 1–14.

[44] Kocher, P., Jaffe, J., and Jun, B. Differential power analysis. *Proceedings of the CRYPTO' 99 1666* (1999), 388–397.

[45] Kocher, P. C. Timing attacks on implementations of diffie-hellman rsa dss and other systems. *Technical report, Cryptography Research. web site http://www.cryptography.com/resources/whitepapers/TimingAttacks.pdf*, 1–10.

[46] Leng, X., Mayes, K., and Markantonakis, K. Hb-mp+ protocol: an improvement on the hb-mp protocol. *Proceedings of the 20008 IEEE International Conference on RFID*, 118–124.

[47] Matsumoto, T., and Imai, H. Human identification through insecure channel. *proceeedings of the EUROCRYPT'91 547* (1991), 409–421.

[48] Meier, W., and Staffelbach, O. Analysis of pseudo random sequences generated by cellular automata. *Advances in Cryptology - Eurocrypt'91 LNCS 547* (1991), 186–199.

[49] MIT. *web sit http://autoid.mit.edu/*.

[50] Molnar, D., and Wagner, D. Privacy and security in library rfid: Issues, practices, and architectures. *proceedings of the ACM Conference on Communications and Computer Security*, 210 – 219.

[51] Munilla, J., and Peinado, A. Hb-mp: A further step in the hb-family of lightweight authentication protocols. *proceedings of the Computer Networks: The International Journal of Computer and Telecommunications Networking 51 , Issue 9* (2007), 2262–2267.

[52] Naor, M., and Pinkas, B. Visual authentication and identification. *Advances in Cryptology - CRYPTO '97 - Lecture Notes in Cimputer Science 1294* (997), 322–336.

[53] Piramuthu, S. Hb and related lightweight authentication protocols for secure rfid tag/reader authentication. *CollECTeR Europe Conference*, 1–8.

[54] Quisquater, J. J., and Couvreur, C. Fast decipherment algorithm for rsa public key cryptosystem. *Electronic Letters 18, Issue 21* (1982), 905 – 907.

[55] Regev, O. On lattices, learning with errors, random linear codes, and cryptography. *37th ACM Symposium on Theory of Computing (STOC 2005)*, 84–93.

[56] Rivest, R. L., Shamir, A., and Aldeman, L. A method for obtaining digital signatures and publickey cryptosystems. *Communications of the ACM 21, No.2* (1978), 120–126.

[57] SafeTzone. *web site http://www.safetzone.com*.

[58] Sarma, S. E. *Technical Report MIT-AUTOID-WH-006, MIT Auto ID Center. web site http://www.epcglobalinc.org*.

[59] Sarma, S. E., Weis, S. A., and Engels, D. W. Radio-frequency-identification security risks and challenges. *CryptoBytes - RSA Laboratories 6(1)* (2003), 1–32.

[60] Takaragi, K., Imura, M. U. R., Itsuki, R., and Satoh, T. An ultra small individual recognition security chip. *IEEE Micro 21(6)* (2001), 43–49.

[61] TECHNOLOGY, S. Where's the smart money? the economist. *web site http://www.ti.com/rfid/docs/news/in-the-news/2002/02-07-02.shtml*, 69–70.

[62] WANG, C.-H., HWANG, T., AND TSAI, J.-J. On the matsumoto and imai's human identification scheme. *proceedings of the EuroCrypt'95 921* (1995), 382–392.

[63] WEIS, S. A., RIVEST, R. L., AND SMITH, A. C. New foundations for efficient authentication, commutative cryptography, and private disjointness testing. *Doctoral Thesis - MASSACHUSETTS INSTITUTE OF TECHNOLOGY - MIT*, 1–115.

[64] WEIS, S. A., SARMA, S. E., RIVEST, R. L., AND ENGELS, D. W. Security and privacy aspects of low-cost radio frequency identification systems. *proceedings of the International Conference on Security in Pervasive Computing - SPC 2003 2802* (2003), 454–469.

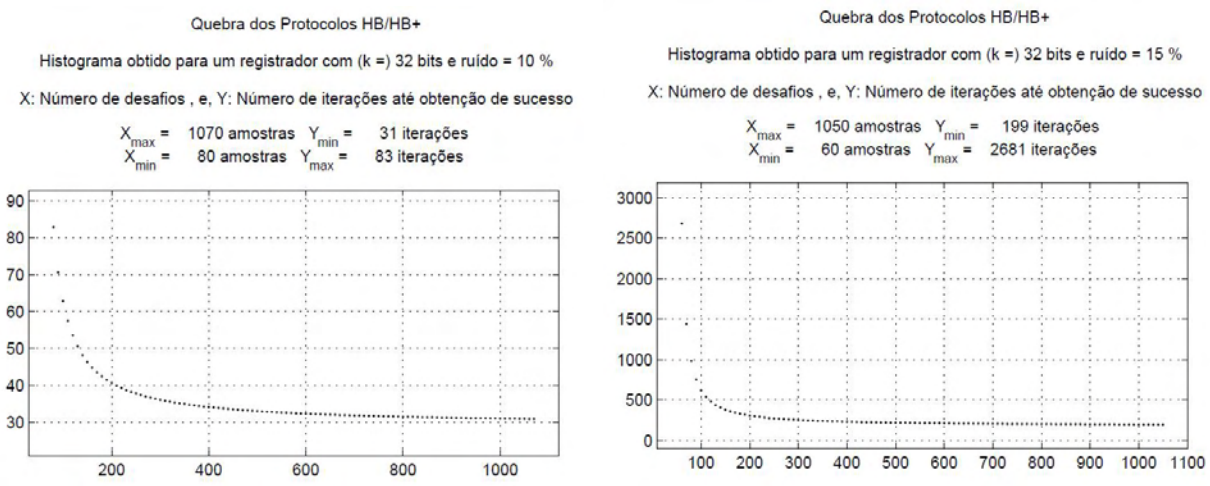
[65] WHITE, D. Ncr: Rfid in retail. *RFID: Applications, Security, and Privacy*, 381–395.

[66] WOLFRAM, S. Cryptography with cellular automata. *Advances in Cryptology - Crypto'85 218* (1986), 429–432.

[67] WOLFRAM, S. Random sequence generation by cellular automata. *Advances in Applied Mathematics, (disponível na website do autor) 7* (1986), 123–169.

ANEXOS

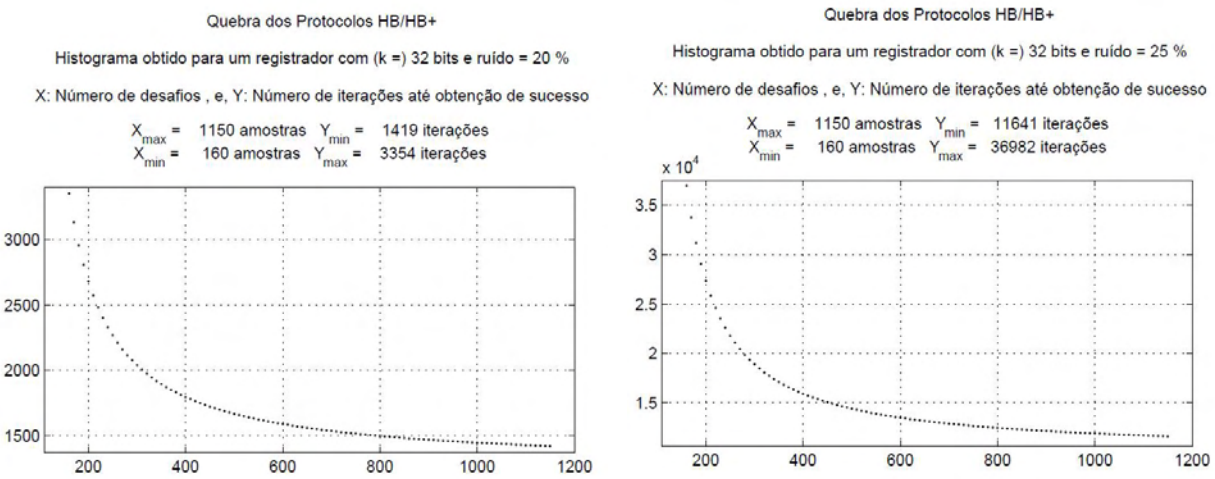
I. GRÁFICOS - ATAQUE PROBABILÍSTICO AOS PROTOCOLOS HB/HB$^+$



(a) k = 32 e $\eta = 10\%$ (b) k = 32 e $\eta = 15\%$

Figura I.1: *Ataque ao Protocolo HB*

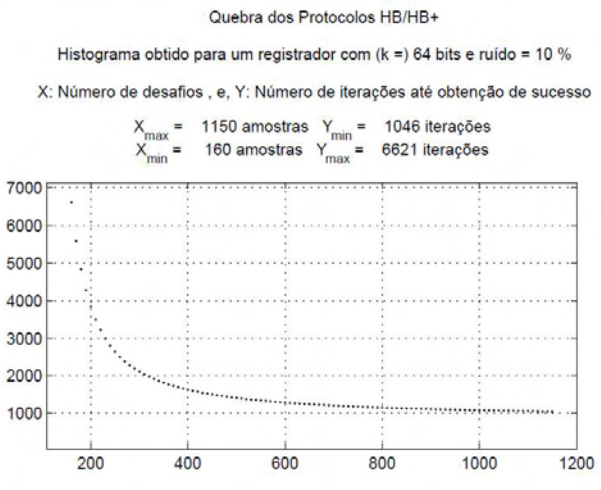
Para a chave com comprimento k = 32, figura I.1, com $\eta = 10\%$, $m = 60$ desafios e 152 iterações; ou com $\eta = 15\%$, $m = 1050$ desafios e 199 iterações.



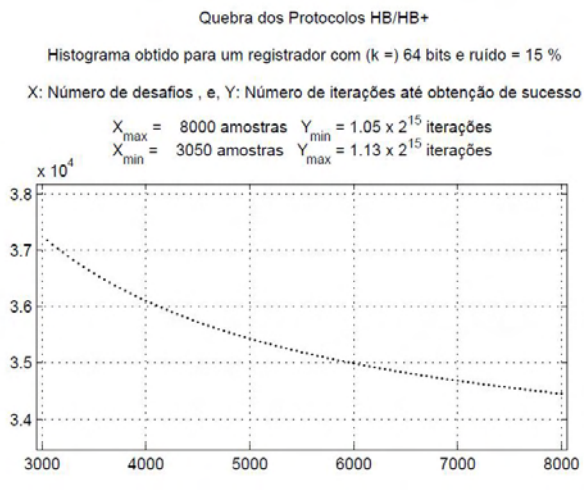
(a) k = 32 e $\eta = 20\%$ (b) k = 32 e $\eta = 25\%$

Figura I.2: *Ataque ao Protocolo HB*

Para a chave com comprimento k = 32, figura I.2,com $\eta = 20\%$, $m = 1050$ desafios e 152 iterações; ou com $\eta = 25\%$, $m = 1050$ desafios e 11641 iterações.



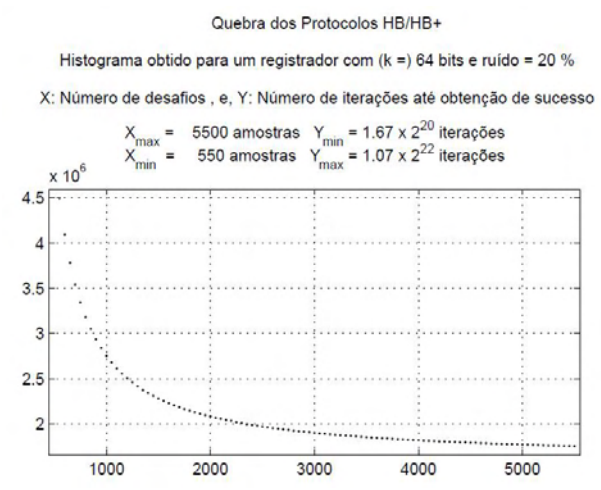
(a) k = 64 e $\eta = 10\%$



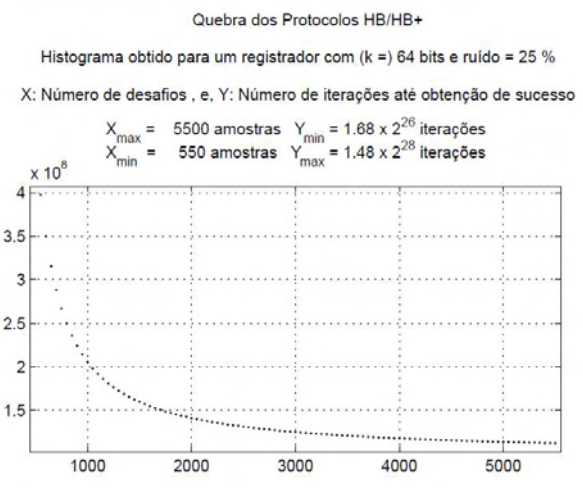
(b) k = 64 e $\eta = 15\%$

Figura I.3: *Ataque ao Protocolo HB*

Para a chave com comprimento k = 64, figura I.3, com $\eta = 10\%$, $m = 2150$ desafios e 215 iterações; ou com $\eta = 15\%$, $m = 5200$ desafios e 219 iterações.



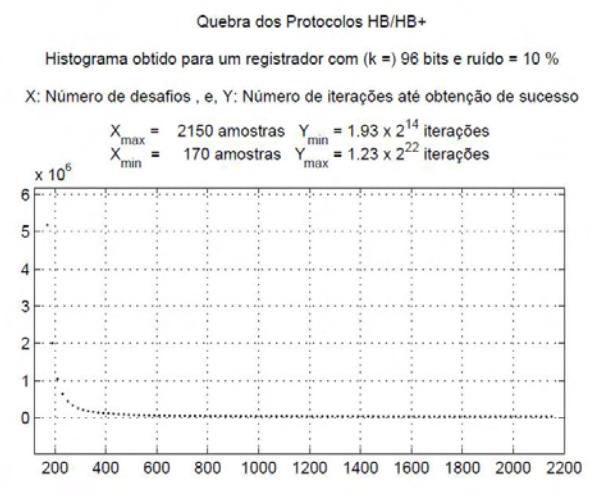
(a) k = 64 e $\eta = 20\%$



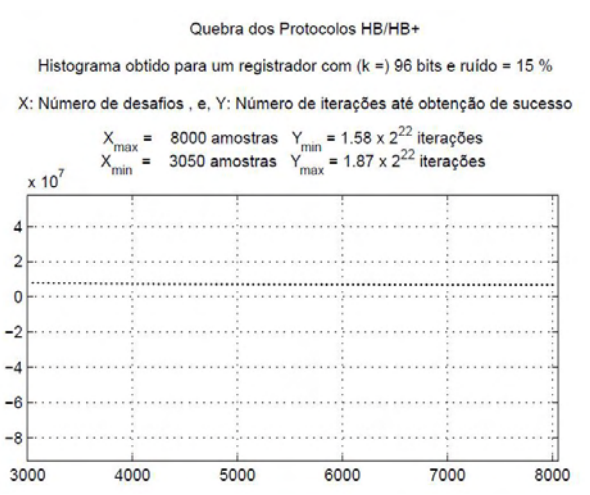
(b) k = 64 e $\eta = 25\%$

Figura I.4: *Ataque ao Protocolo HB*

Para a chave com comprimento k = 64, figura I.4, com $\eta = 20\%$, $m = 5200$ desafios e 223 iterações, ou com $\eta = 25\%$, $m = 5600$ desafios e 240 iterações.



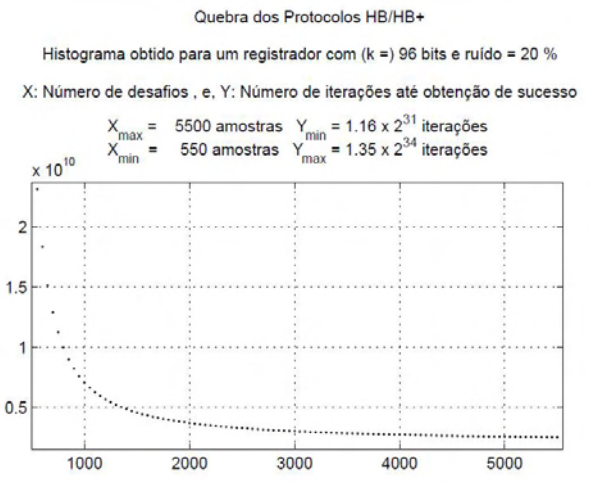
(a) k = 96 e $\eta = 10\%$



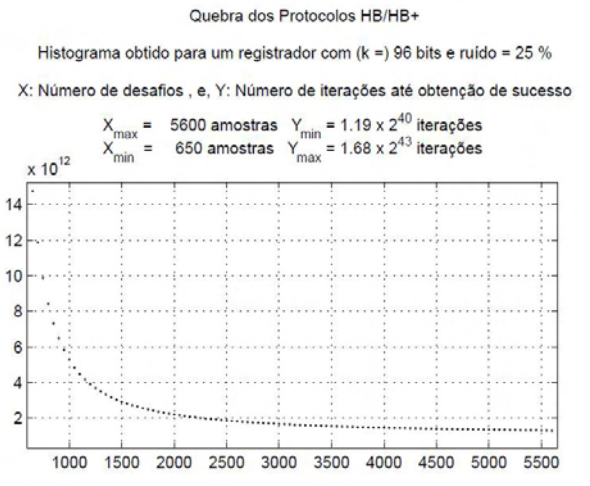
(b) k = 96 e $\eta = 15\%$

Figura I.5: *Ataque ao Protocolo HB*

Para a chave com comprimento k = 96, figura I.5, com $\eta = 10\%$, $m = 2150$ desafios e $\approx 2^{15}$ iterações; ou com $\eta = 15\%$, $m = 5200$ desafios e $\approx 2^{19}$ iterações.



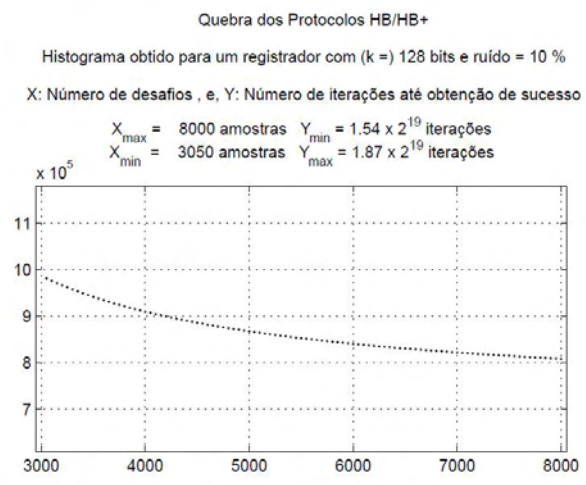
(a) k = 96 e $\eta = 20\%$



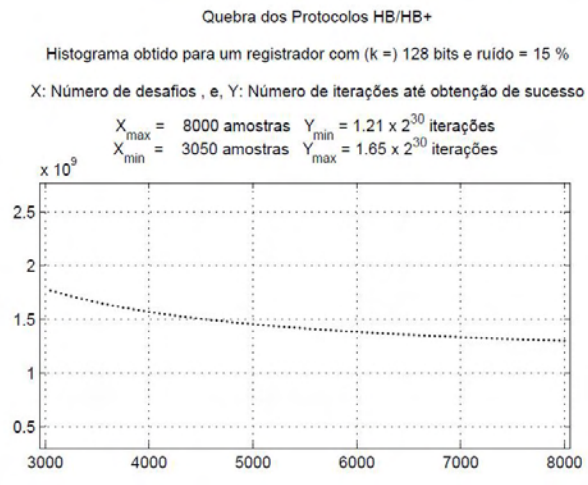
(b) k = 96 e $\eta = 25\%$

Figura I.6: *Ataque ao Protocolo HB*

Para a chave com comprimento k = 96, figura I.6, com $\eta = 20\%$, $m = 5200$ desafios e $\approx 2^{23}$ iterações, ou com $\eta = 25\%$, $m = 5600$ desafios e $\approx 2^{40}$ iterações.



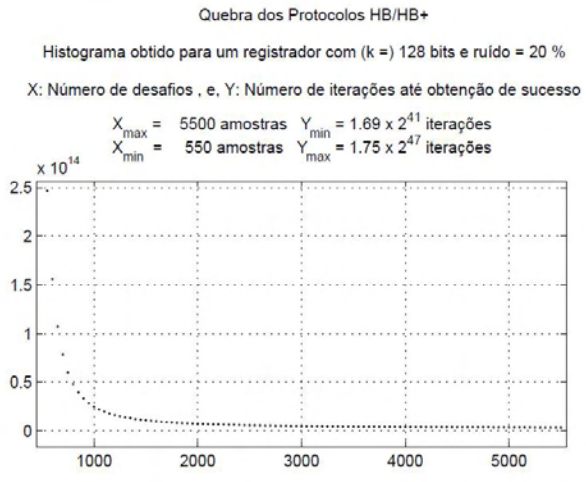
(a) k = 128 e $\eta = 10\%$



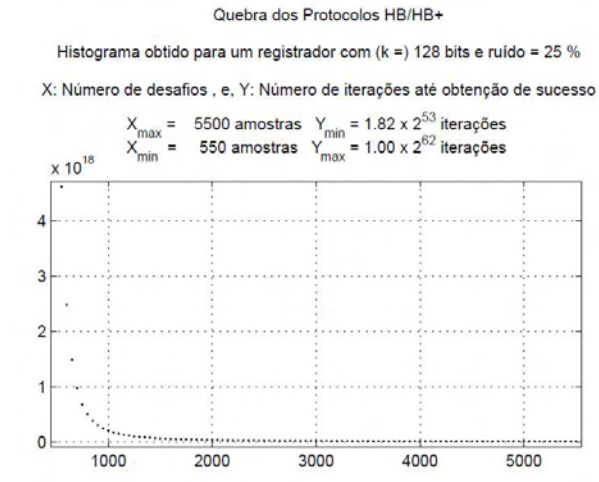
(b) k = 128 e $\eta = 15\%$

Figura I.7: *Ataque ao Protocolo HB*

Para a chave com comprimento k = 128, figura I.7, com $\eta = 10\%$, $m = 5500$ desafios e 852329 iterações; ou com $\eta = 15\%$, $m = 550$ desafios e $\approx 2^{35}$ iterações, ou, $m = 5500$ desafios e $\approx 2^{30}$ iterações.



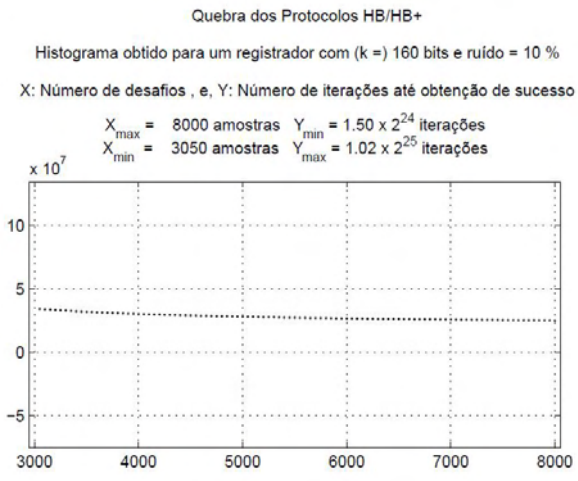
(a) k = 128 e $\eta = 20\%$



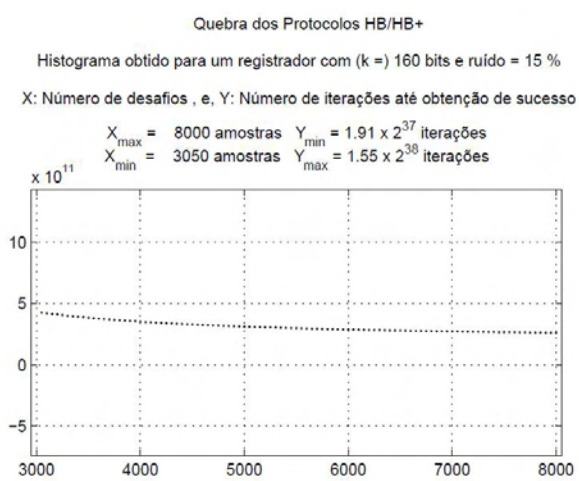
(b) k = 128 e $\eta = 25\%$

Figura I.8: *Ataque ao Protocolo HB*

Para a chave com comprimento k = 128, figura I.8, com $\eta = 20\%$, $m = 5500$ desafios e $\approx 2^{38}$ iterações; ou com $\eta = 25\%$, $m = 5500$ desafios e $\approx 2^{54}$ iterações.



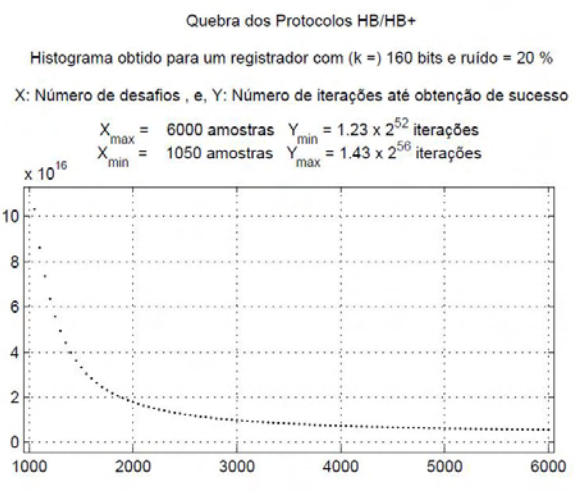
(a) k = 160 e $\eta = 10\%$



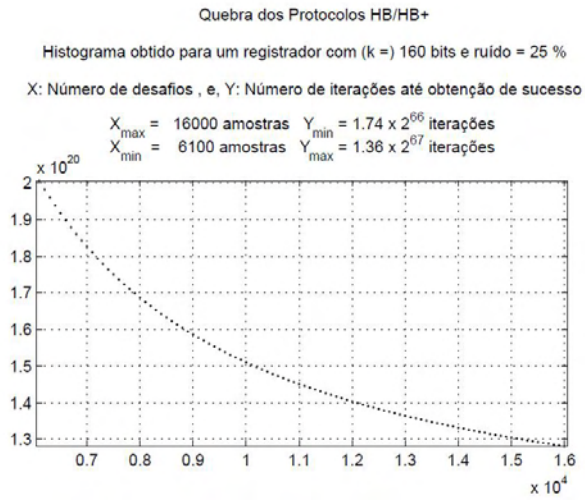
(b) k = 160 e $\eta = 15\%$

Figura I.9: *Ataque ao Protocolo HB*

Para a chave com comprimento k = 160, figura I.9, com $\eta = 10\%$, $m = 550$ desafios e $\approx 2^{29}$ iterações; ou com $\eta = 15\%$, $m = 5500$ desafios e $\approx 2^{38}$ iterações.



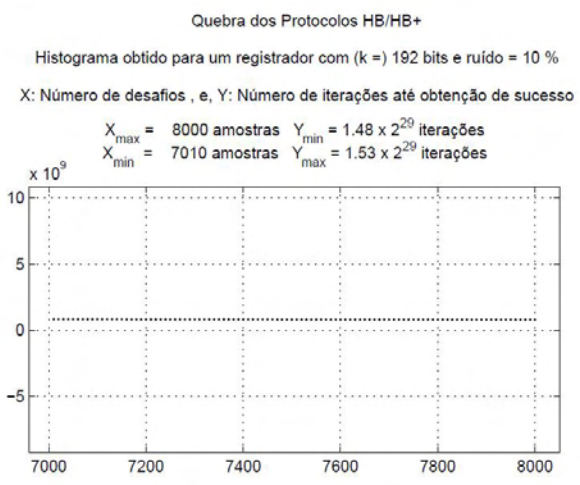
(a) k = 160 e $\eta = 20\%$



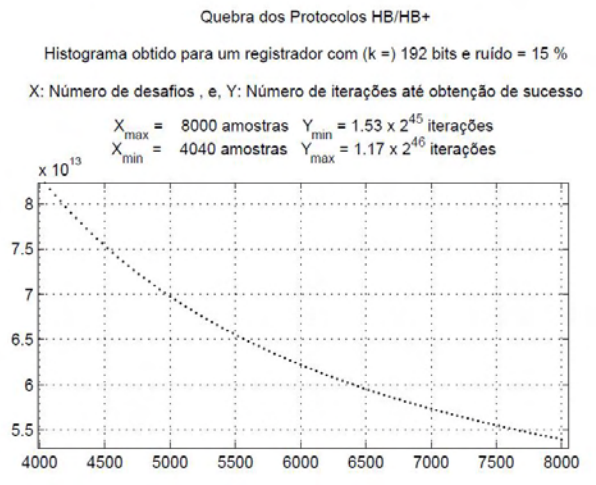
(b) k = 160 e $\eta = 25\%$

Figura I.10: *Ataque ao Protocolo HB*

Para a chave com comprimento k = 160, figura I.10, com $\eta = 20\%$, $m = 6000$ desafios e $\approx 2^{52}$ iterações; ou com $\eta = 25\%$, $m = 12000$ desafios e $\approx 2^{67}$ iterações.



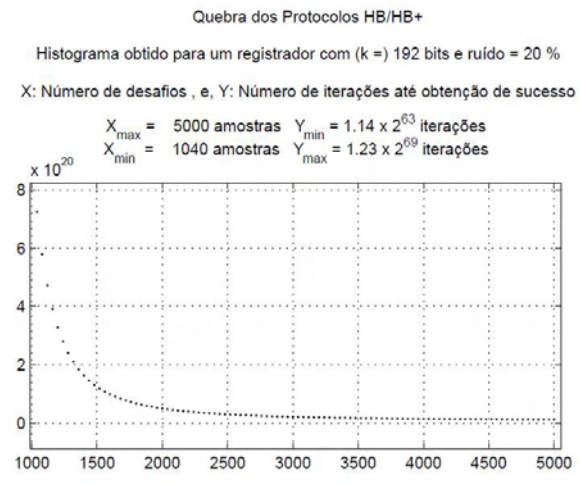
(a) k = 192 e $\eta = 10\%$



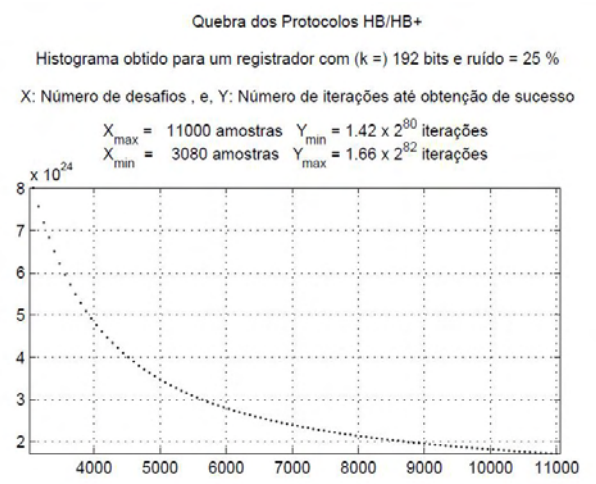
(b) k = 192 e $\eta = 15\%$

Figura I.11: *Ataque ao Protocolo HB*

Para a chave com comprimento k = 192, figura I.11, com $\eta = 10\%$, $m = 1500$ desafios e $\approx 2^{31}$ iterações; ou com $\eta = 15\%$, $m = 5500$ desafios e $\approx 2^{46}$ iterações.



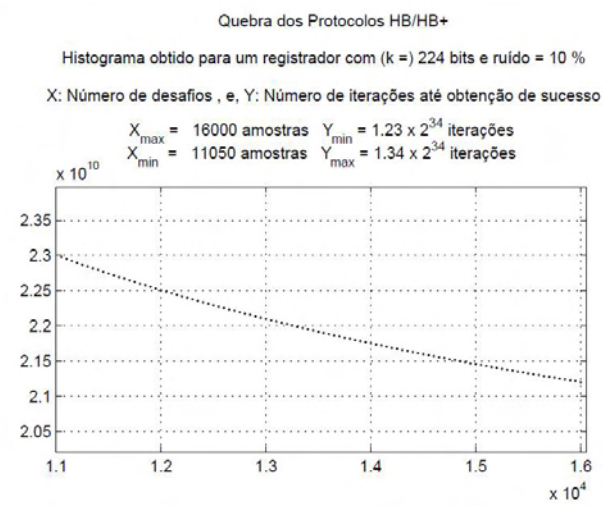
(a) k = 192 e $\eta = 20\%$



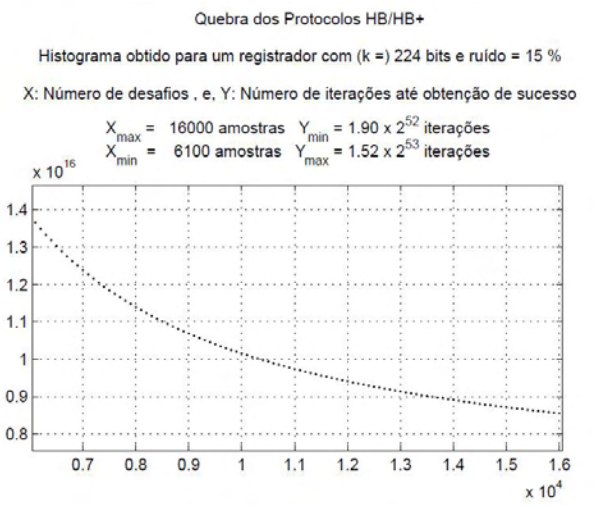
(b) k = 192 e $\eta = 25\%$

Figura I.12: *Ataque ao Protocolo HB*

Para a chave com comprimento k = 192, figura I.12, com $\eta = 20\%$, $m = 5000$ desafios e $\approx 2^{63}$ iterações; ou com $\eta = 25\%$, $m = 11000$ desafios e $\approx 2^{80}$ iterações.



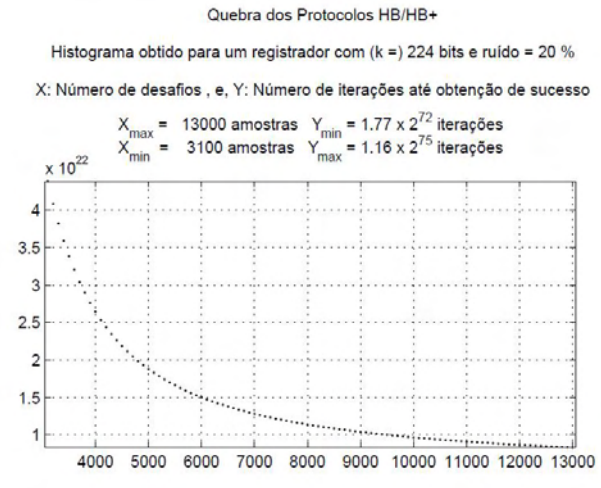
(a) k = 224 e $\eta = 10\%$



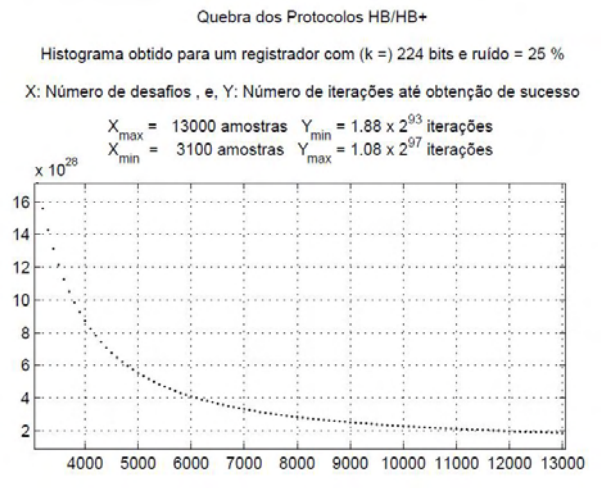
(b) k = 224 e $\eta = 15\%$

Figura I.13: *Ataque ao Protocolo HB*

Para a chave com comprimento k = 224, figura I.13, com $\eta = 10\%$, $m = 6000$ desafios e $\approx 2^{35}$ iterações; ou com $\eta = 15\%$, $m = 3100$ desafios e $\approx 2^{55}$ iterações.



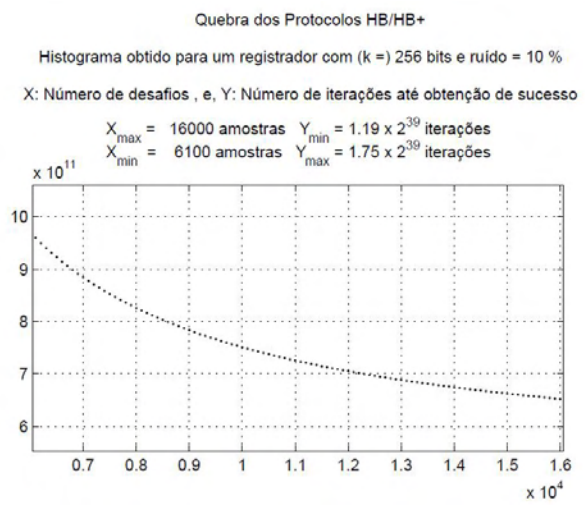
(a) k = 224 e $\eta = 20\%$



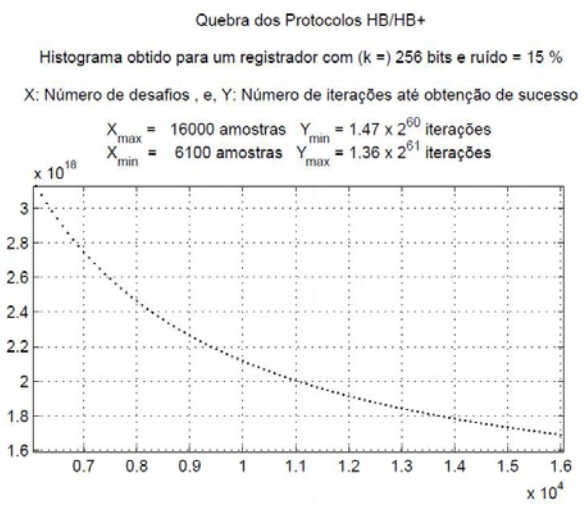
(b) k = 224 e $\eta = 25\%$

Figura I.14: *Ataque ao Protocolo HB*

Para a chave com comprimento k = 224, figura I.14, com $\eta = 20\%$, $m = 3100$ desafios e $\approx 2^{75}$ iterações; ou com $\eta = 25\%$, $m = 13000$ desafios e $\approx 2^{93}$ iterações.



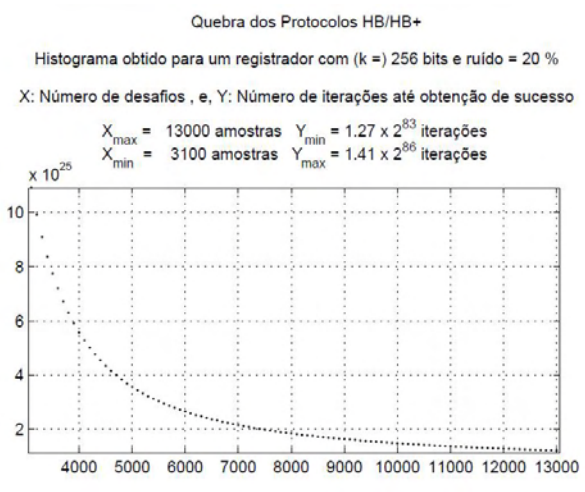
(a) k = 256 e $\eta = 10\%$



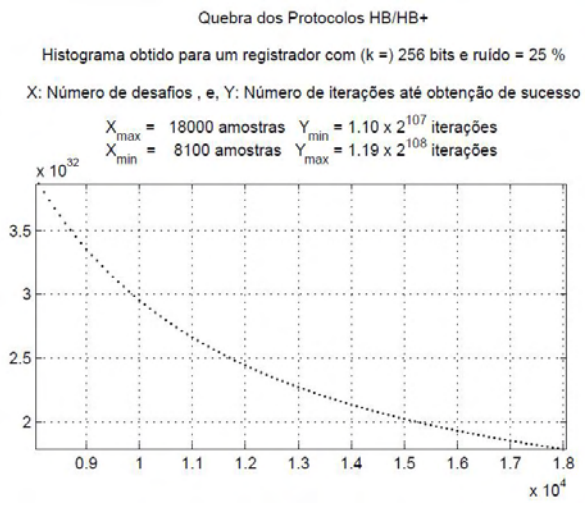
(b) k = 256 e $\eta = 15\%$

Figura I.15: *Ataque ao Protocolo HB*

Para a chave com comprimento k = 256, figura I.15, com $\eta = 10\%$, $m = 3100$ desafios e $\approx 2^{40}$ iterações; ou com $\eta = 15\%$, $m = 13000$ desafios e $\approx 2^{60}$ iterações.



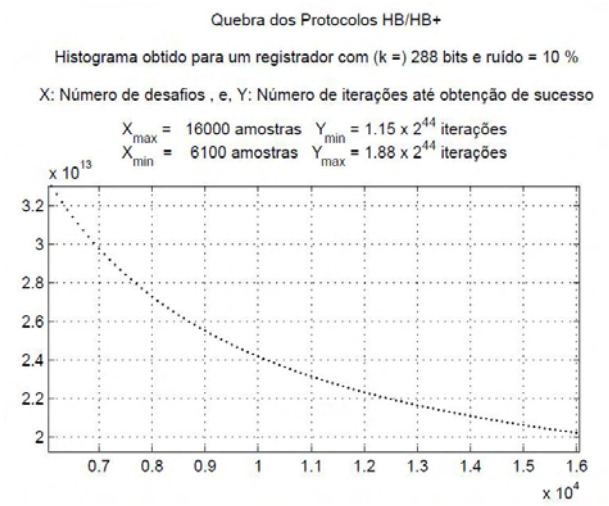
(a) k = 256 e $\eta = 20\%$



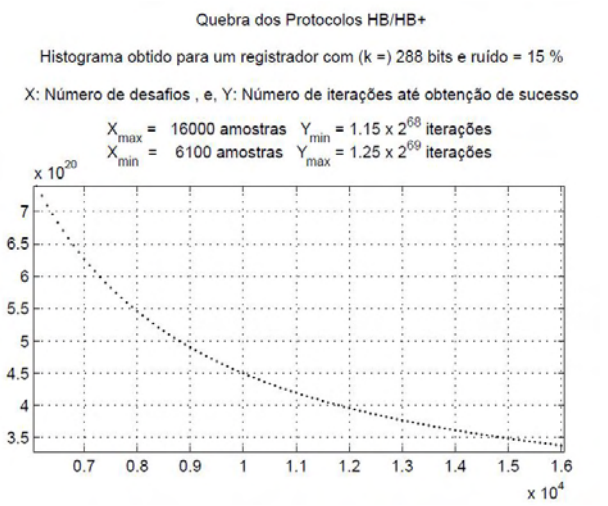
(b) k = 256 e $\eta = 25\%$

Figura I.16: *Ataque ao Protocolo HB*

Para a chave com comprimento k = 256, figura I.16, com $\eta = 20\%$, $m = 13000$ desafios e $\approx 2^{83}$ iterações, ou com $\eta = 25\%$, $m = 18000$ desafios e $\approx 2^{107}$ iterações.



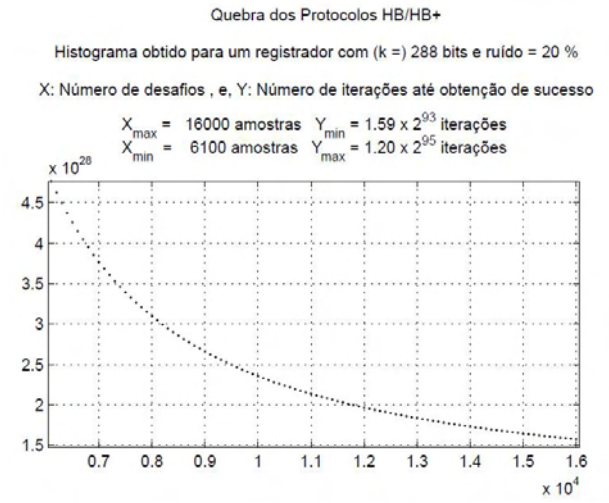
(a) k = 288 e $\eta = 10\%$



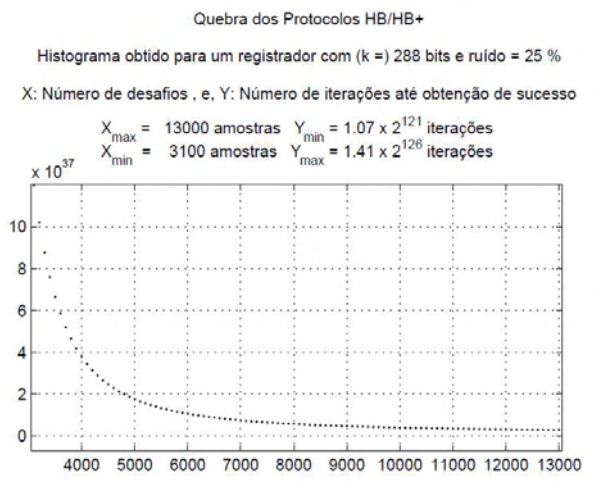
(b) k = 288 e $\eta = 15\%$

Figura I.17: *Ataque ao Protocolo HB*

Para a chave com comprimento k = 288, figura I.17, com $\eta = 10\%$, $m = 3100$ desafios e $\approx 2^{46}$ iterações; ou com $\eta = 15\%$, $m = 13000$ desafios e $\approx 2^{68}$ iterações.



(a) k = 288 e $\eta = 20\%$

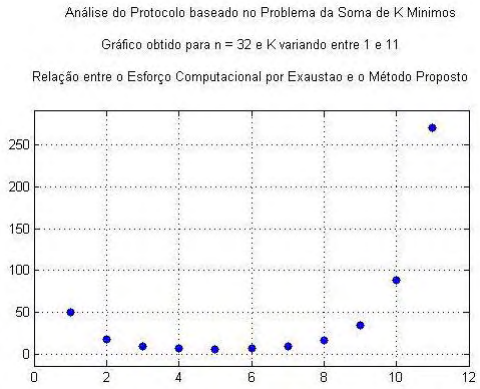


(b) k = 288 e $\eta = 25\%$

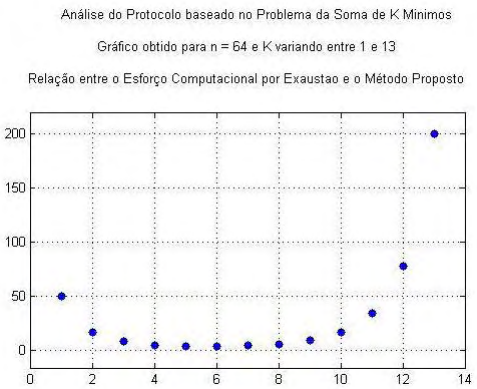
Figura I.18: *Ataque ao Protocolo HB*

Para a chave com comprimento k = 288, figura I.18, com $\eta = 20\%$, $m = 13000$ desafios e $\approx 2^{94}$ iterações; ou com $\eta = 15\%$, $m = 13000$ desafios e $\approx 2^{121}$ iterações.

II. GRÁFICOS - PROBLEMA DA SOMA DE K MÍNIMOS

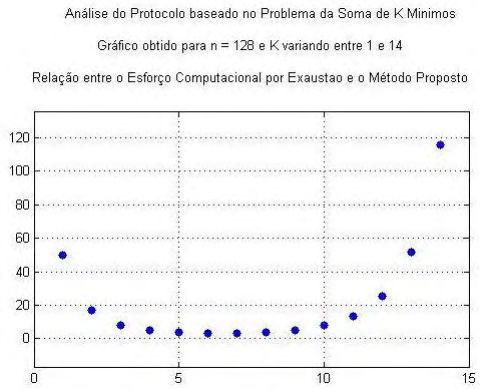


(a) n = 32 e k = 1 a 11

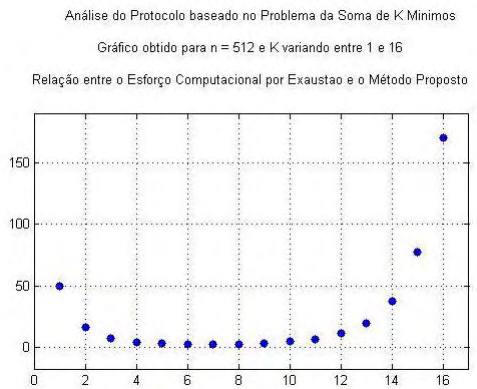


(b) k = 64 e k = 1 a 13

Figura II.1: *Relação de Complexidade: Por Exaustão × Método Passivo*

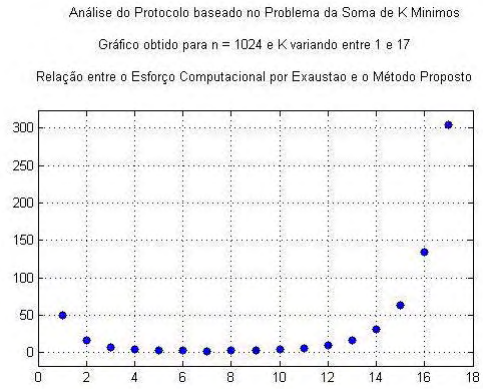


(a) n = 128 e k = 1 a 14



(b) k = 512 e k = 1 a 16

Figura II.2: *Relação de Complexidade: Por Exaustão × Método Passivo*



(a) n = 1024 e k = 1 a 17

Figura II.3: *Relação de Complexidade: Por Exaustão × Método Passivo*

III. PSEUDO-CÓDIGO DO ATAQUE ATIVO A PROTOCOLOS BASEADOS NA SOMA DE K MÍNIMOS

Descrição minuciosa do nosso método de ataque ativo a Protocolos baseados no Problema da Soma de K:

Parte 1: determinam-se quais são os números inteiros x_j e $y_j \in z$

1. Para $j = 0$, gerar v_j tal que:
   (a) $v_{ij} = r$, $2 \leq r < n$, e, $i = 1, n$
   (b) Calcular o desafio u_j, equação 5.1; neste caso equivale a $t = k \times r \ mod \ n$
2. $j = 0$;
3. Gerar v_{ij}:
   (a) $i = 0$
   (b) se $i \neq j$, $v_{ij} = r \ mod \ n$, para $r \geq 2$
   (c) se $i = j$, $v_{ij} = r - 1 \ mod \ n$, para $r \geq 2$
   (d) $i = i + 1$
   (e) se $i < n$ vá para $3(b)$
4. Calcular u_j, equação 5.1, fazer $w_j = u_j$
5. j = j + 1
6. se $j < n$ vá para 3
7. Análise de w_j para se determinar quais inteiros pertencem a chave z:
   (a) $j = 0$
   (b) se $w_j = t$ (item $1(b)$) $\Rightarrow j \notin z$
   (c) se $u_j = (t - S) \ mod \ n$, j tem S ocorrências em z
   (d) $j = j + 1$
   (e) se $j < n$ vá para $7(b)$

Observação: a variável w_j é utilizada apenas para guardar a informação u_j

Parte 2: determinam-se quais são os pares $(x_p, y_p) = (y_p, x_p) \in z$, para $p = 1, k$:

1. $j = 0$

2. $v_{ij} = 0$, para $i = 0, n$

   (a) $i = 0$

   (b) se $w_i = t \Rightarrow q_i = n$ (n nunca ocorre), vá para $2(d)$

   (c) se $w_i \neq t \Rightarrow q_i = 0$

   (d) $i = i + 1$, se $i < n$, vá para $2(b)$

3. $p = j$

4. se $q_p = n$, vá para $4(h)$

   (a) $v_{pj} = r \ mod \ n$

   (b) $p = p + 1$, se $p = n$, vá para $4(h)$

   (c) se $q_p = n$, vá para $4(b)$

   (d) $v_{pj} = r - 1 \ mod \ n$

   (e) Calcular u_p, equação 5.1, fazer $q_p = u_p$

   (f) $v_{pj} = 0$

   (g) vá para $4(b)$

   (h) $p = p + 1$, se $p < n$, vá para 4

5. se $q_p = r - 1$, o par $(j, p) = (p, j) \in z$, para quaisquer $p = j + 1, n - 1$

6. para $q_p = r - 1$, fazer $q_p = 0$, para quaisquer $p = j + 1, n - 1$

7. $j = j + 1$, se $j < n$, vá para 3

Observação: a variável w_j é utilizada apenas para guardar a informação u_j