Criptografia com bloco de correção de erros aplicados à evolução de autômatos

Anderson P. A. Chaves [1], Allex S. Albuquerque [2], Bruno Sokal [3], Francisco J. A. Aquino [4]

Instituto Federal de Educação, Ciência e Tecnologia do Ceará

[1] andersonchaves.eng@gmail.com, [2] allex.silveira.ce@gmail.com, [3] brunosokal@gmail.com, [4] fcoalves_aq@ifce.edu.br

Departamento de Telemática

Av. 13 de Maio, Nr. 2081, Campus de Fortaleza

Fortaleza/CE, Brasil, 60040-531

Resumo – este artigo busca apresentar uma técnica de criptografia de mensagens digitais utilizando pares de regras (uma regra para encriptação e outra para decriptação) de autômatos celulares perfeitamente invertíveis e imperfeitamente invertíveis, e um código de controle de erros, no qual terá como principal objetivo tornar as regras ditas como imperfeitamente invertíveis em regras "perfeitas". Foi desenvolvido também um *software* com dois aplicativos, um para encriptação de dados, onde o usuário insere uma mensagem (em ASCII) desejada a ser codificada e outra para a decriptação de dados, onde se terá a mensagem original.

I. Introdução

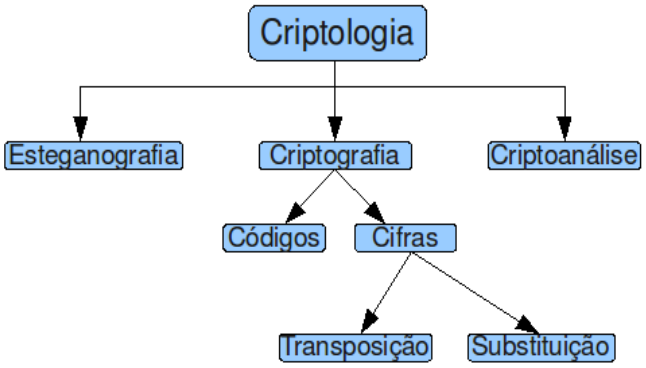
A comunicação entre máquinas tornou-se fundamental e indispensável. Relatórios indicam que sua importância aumentará cada vez mais no futuro [1], por isso, novas técnicas continuam a surgir para tentar melhorar, cada vez mais, o desempenho da comunicação entre máquinas e manter o sigilo dos dados.

Quando falamos em segurança e criptografia, devemos ter em mente de que nenhum sistema é totalmente seguro. Grande parte das técnicas de criptografia utilizadas hoje em grande parte do mundo está baseada em dificuldades e restrições tecnológicas e de processamento dos computadores atuais, como a RSA e a Diffie-Hellman. Porém, pesquisadores já advertem que estas tecnologias criptográficas estão cada vez menos eficientes, e que talvez dentro de quatro ou cinco anos elas possam ser desfeitas [2]. Surge portanto a necessidade de investir na aplicação de novas tecnologias como solução para a criptografia clássica.

Alternativas já estão surgindo, como a Criptografia Quântica [3], a Criptografia Caótica [4] [5], e a Criptografia Neurocientífica [6]. Dentre todas essas técnicas, a Criptografia Caótica ocupa um papel importante enquanto técnica alternativa de criptografia, sendo um ramo de criptografia atualmente em ascensão, e a semente inicial para as promissoras técnicas criptográficas [7] [8]. Assim, justifica-se o estudo de criptografias com autômatos celulares neste trabalho por serem eles ferramentas muito úteis e práticas para a implementação de tais métodos.

II. Criptologia e Criptografia

Dá-se o nome de Criptologia ao ramo de estudo responsável por estudar as diversas formas de se codificar e de se decodificar informações ou mensagens. A criptologia possivelmente teve início com o surgimento da escrita, mas sua utilização foi primeiramente identificada e documentada com os egípcios em cerca de 1900 a 2000 a.C.. Apesar disso, foi durante as duas Guerras Mundiais que a criptologia passou por grandes avanços em seus estudos, com o surgimento de importantes máquinas de codificação da história, como a Enigma, criada pelos nazistas na Segunda



Mundial, as praias de desembarque das tropas eram conhecidas pelos códigos Omaha, Juno, etc.), deste modo pode-se dizer que um código manipula o significado da

mensagem, ao invés da cifra que funciona como uma alteração da representação da mensagem. Um exemplo clássico, será o cifrar da palavra “cubo” para “dvcp”, em que apenas se substituiu cada letra pela seguinte do alfabeto (cifra usada por Júlio César, ver Fig. 1). Logo após passar pelo processo de cifragem, a mensagem “com significado obscuro” é conhecida como texto cifrado, ou criptograma. O processo contrário, ou seja, retornar uma mensagem irreconhecível para sua forma original novamente é chamado de decriptação (decodificação ou decifragem). A cifra pode ser de substituição, quando os caracteres da mensagem são trocados por outros a partir de uma lógica, ou de transposição, quando os caracteres da mensagem são mantidos, mas mudam de posição.

Chamamos de chave o parâmetro de segurança de operação da cifra, pois para cada chave diferente fornecida à cifra, um novo criptograma poderá ser gerado.

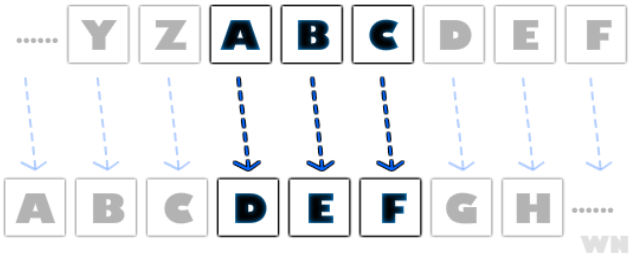


Fig. 1 – Exemplificando a Cifra de César.

A Fig. 2 mostra as ramificações da criptologia. Ao ramo da criptologia responsável por estudar e aplicar métodos diversos de encriptação ou cifragem chamamos Criptografia. Ao ramo da criptologia responsável por estudar e aplicar métodos de decriptação ou decifragem chamamos Criptoanálise. Há ainda outro ramo da criptologia conhecido como Esteganografia, que se preocupa em estudar e aplicar métodos que visam esconder a própria existência de uma mensagem, por exemplo, usando tinta invisível, ou destacando discretamente algumas letras numa mensagem arbitrária.

Fig. 2 – Diagrama mostrando as subdivisões de estudo dentro da criptologia.

Os algoritmos modernos utilizam uma chave para controlar a codificação e decodificação da mensagem. Tipicamente, a codificação de uma mensagem M possui um emissor chamado Alice, e um receptor chamado de Bob. A mensagem M é escrita em um alfabeto A, e cada símbolo do alfabeto pode ser mapeado para um valor inteiro. Então, para enviar a mensagem:

ALICE:

- Converte a mensagem M, que é uma sequência de caracteres, em uma sequência de inteiros;

  MensagemParaInteiros(M)
- Criptografa essa mensagem, mapeando essa sequência de inteiros em uma outra sequência de inteiros, resultando na mensagem criptografada;

  E <-- Criptografar(MensagemParaInteiros(M))
- Envia E para o receptor Bob.

BOB:

- Decripta a mensagem, recuperando a sequência de inteiros original;

  Decriptar(E)
- Reconverte a sequência de inteiros ao texto original M.

  M <-- InteirosParaMensagem(Decriptar(E))

Para que o algoritmo seja eficaz, as funções devem possuir a propriedade de uma ser o inverso da outra. Além de que, Alice deve computar Criptografar eficientemente, enquanto que uma pessoa não autorizada não deve ser capaz de computar (Decriptar) eficientemente. Entende-se eficiência por rapidez.

Existem funções com essas propriedades, que são chamadas funções *trapdoor*. Elas são fáceis de computar, mas difíceis de inverter, a menos que se conheça ou se tenha acesso a uma chave secreta K.

Então, para criptografar uma mensagem M Alice computa:

E <-- Criptografar(MensagemParaInteiros(M),K))

E para recuperar a mensagem, Bob computa:

M <-- InteirosParaMensagem(Decriptar(E,K))

De acordo com a chave que utiliza, os algoritmos clássicos podem ser:

A. Simétricos ou de Chave Privada

Utilizam uma única chave tanto para criptografar quanto decriptar a mensagem. Desse modo, os envolvidos na comunicação precisam possuir a mesma chave e esta deve ser secreta (ver Fig. 3). Para que este mecanismo possa funcionar com segurança é necessária a existência de um canal seguro para a transmissão da chave, e esta deve ser trocada a cada nova comunicação, caso contrário está sujeita a ser quebrada por criptoanálise. Um exemplo de algoritmo simétrico é o One Time Pad.

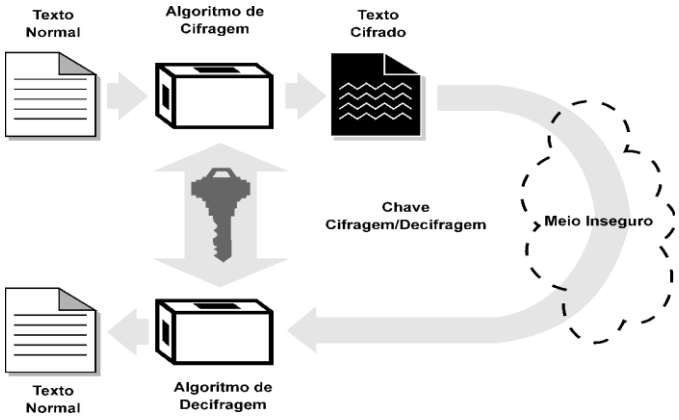


Fig. 3 – Esquema de criptografia de chave privada.

B. Assimétricos ou de Chave Pública

Utilizam duas chaves no processo (ver Fig. 4): a chave pública para criptografar a mensagem, e a chave privada para decriptar. Não havendo troca de chaves. Seu exemplo mais conhecido é o RSA..

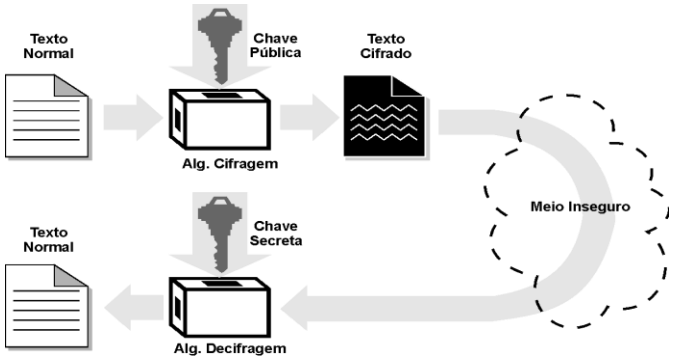


Fig. 4 – Esquema de criptografia de chave pública.

III. Autômatos Celulares

Autômato celular é uma ferramenta bastante utilizada para descrever sistemas evolutivos discretos [9], no qual o valor de cada célula depende do valor da sua vizinhança num tempo anterior. Eles têm sido amplamente estudados em diversas áreas da ciência, como por exemplo, a utilização de autômatos celulares para estimar as áreas de serviço das subestações de energia elétrica [10], processo de formação de cristais de gelo [11], no estudo de espalhamentos de epidemias [12], entre outros.

Um autômato celular pode ser definido como um conjunto de células com determinados valores e para cada célula existe um conjunto de células normalmente chamadas de vizinhança (na maioria dos casos incluindo a própria célula em si) que interagem entre si em função de uma coleção finita de condições pré-definidas. Os valores, ou estados, das células são alterados conforme um conjunto de regras de transição, que dependem da vizinhança. No estado inicial (t), os valores para cada célula são definidos. Após uma evolução (t+1), é criada uma nova geração de células com valores definidos pela regra de transição aplicada á vizinhança. As bordas são podem ser tratadas de formas diferentes, no presente trabalho são tratadas de acordo com a Fig. 5.

Podemos representar a composição de um AC como quatro partes $Y = (L,p,k, f)$, onde o L é conhecido como lattice do autômato e representa o seu formato ou forma geométrica, p é o conjunto de estados que pode ser assumido por cada célula, k é a vizinhança de uma determinada célula (fator influenciado pelo raio r definido para o autômato), e o f a função de transição de estados ou regra de transição de estados (MELOTTI, 2009). Para um autômato celular unidimensional que possa assumir dois valores, sua vizinhança é $k = 2r + 1$, onde r é o raio da célula, há $2k$ combinações de vizinhos e 2^{2^k} regras, ou seja, nesse caso 256 regras. De um modo geral, a regra de transição de estados é imposta de forma paralela e sincronizada em todas

as células, o que caracteriza um autômato celular uniforme. Um autômato unidimensional uniforme de lattice quadrada, r = 1, k = 3, p = 2, evoluindo com a regra 90 teria como representação da sua função de evolução f tal como representado na Fig. 5. As regras utilizadas por este trabalho (funções p) estão na Tabela 1. Os pares de regras citados como perfeitos, são pares utilizados na encriptação e decriptação com sucesso, já os pares indicados como imperfeitos são pares em que fora preciso utilizar blocos de controle de erros para que sua encriptação e decriptação ocorra com sucesso.

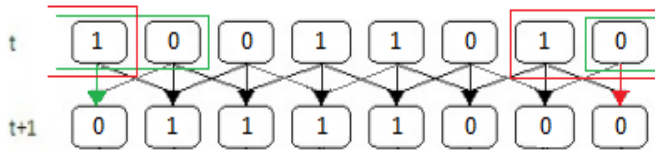


Fig. 5 – Representação de um autômato celular unidimensional evoluindo com a regra 90.

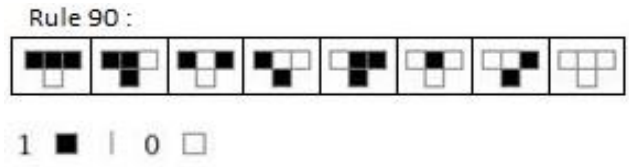


Fig. 6 – Representação da regra 90 de evolução para um autômato celular unidimensional.

IV. Códigos Detectores e Corretores de Erros

A função de um sistema de comunicação é transmitir informação (sinais de banda base) de um lugar para outro através de um canal. Devido a diversos fatores, tais como o ruído (sinal elétrico não desejável) a informação pode chegar ao receptor com erros. O controle de erros pode ser exercido basicamente por duas formas: um sistema FEC (Forwad Error Correction) que consiste em adicionar bits de redundância de acordo com alguma regra pré-estabelecida, para que o receptor ao decodificar a mensagem consiga distinguir quais bits de mensagem foram efetivamente transmitidos. Ou por um sistema ARQ (Automatic Repeat Request) que também utiliza a redundância de bits para detecção de erros, e após a detecção, o receptor solicita o reenvio da informação. Existem algumas estratégias FEC, tais como: Código de Blocos, Códigos Convolucionais, Códigos Turbo [13]. Foi utilizado neste trabalho um código de bloco.

A. Código de Bloco

A mensagem é subdividida em blocos sequenciais, cada um com k bits, e cada bloco de k bits é mapeado em blocos de n bits em que $n > k$. A quantidade de bits adicionais para a redundância equivale a $n - k$. Assim, a notação usada para designar o código de bloco é (n,k). Sendo r_c a taxa de código, então é dada pela equação (1) abaixo:

$$r_c = k / n \tag{1}$$

O código de controle de erros que foi utilizado é um código de bloco linear (8,4), ou seja, a mensagem possui um tamanho de oito bits e são usados quatro bits para redundância. Sua taxa de código é meio (0,5), isso indica a quantidade de informação que a mensagem carrega, que no caso do código de controle usado são quatro bits. Um código de bloco é linear se duas palavras-código quaisquer de um código puderem ser somadas em aritmética módulo 2 para gerar um terceira palavra desse mesmo código.

V. Metodologia e Descrição da Técnica Proposta

A técnica desenvolvida neste artigo funciona atualmente para caracteres de codificação ASCII estendida. Segue abaixo a descrição em detalhes da técnica desenvolvida.

A. Pré-processamento e código corretor de erros

O pré-processamento consiste em preparar a mensagem para que o algoritmo de correção de erros corrija os erros que são introduzidos pelo uso de regras não invertíveis, ao mesmo tempo é uma forma de codificar a mensagem original. Essa etapa consiste na inserção de caracteres extras que são intercalados na mensagem.

Após o pré-processamento da mensagem original é feita uma adição de bits de paridade (redundância) para que se possa fazer uma correção de dados posteriormente.

B. Evolução no autômato celular

Após a adição de bits de paridade, a mensagem se encontra pronta para ser criptografada. Nesse ponto, a mensagem é vista como um conjunto de bits que forma um autômato celular, estando sujeita a regras de evolução por n ciclos, com $n \geq 0$.

Somente um par de regras imperfeitas pode ser utilizado, e apenas uma vez no processo de codificação, pois o algoritmo corretor de erros não teria capacidade de corrigir os erros gerados pelo uso de mais de uma regra desse tipo.

No processo de decodificação, o autômato evolui novamente n vezes de acordo com as regras complementares às utilizadas no processo de criptografia (ver Tabela I).

Foi observado em nosso estudo que, no processo de decodificação, a ordem de utilização das regras complementares é comutativa, isto é, não altera o resultado final da mensagem.

C. Pós-processamento e correção de erros

A mensagem resultante segue para o algoritmo de correção de erros. A etapa do pós-processamento consiste na remoção dos caracteres extras adicionados durante a etapa de pré-processamento.

TABELA I – RELAÇÃO DE PARES DE REGRAS.

PARES DE REGRAS PERFEITAMENTE INVERTÍVEIS	PARES DE REGRAS CORRIGIDAS
15 ↔ 85	43 ↔ 113
51 ↔ 51	142 ↔ 212
170 ↔ 240	
204 ↔ 204	

Fonte – os autores. Fortaleza, Agosto de 2013

VI. APRESENTAÇÃO DE RESULTADOS

O software desenvolvido neste trabalho consiste em dois aplicativos, um para codificar a mensagem e outro para decodificá-la.

A interface do software codificador consiste em uma caixa de texto para a inserção da mensagem a ser processada, caixas de texto onde podem ser inseridas as regras de evolução ao autômato, bem como caixas indicando a quantidade de vezes que cada uma delas pode ser aplicada. Uma caixa de texto também mostra o texto codificado (ver Fig. 7).

O aplicativo gera um arquivo com a mensagem criptografada é gerado no diretório raiz de onde é executado o aplicativo codificador. Essa mensagem criptografada pode ser enviada pelo meio de comunicação desejado (por exemplo, o e-mail).

Já no aplicativo de decodificação, a interface também consiste de caixas de texto onde devem ser informadas as mesmas regras utilizadas na encriptação, pois as regras complementares de reversão serão automaticamente associadas e aplicadas (Fig. 8). O aplicativo lê a mensagem codificada gerada pelo aplicativo codificador e reproduz a mensagem original em uma caixa de texto.

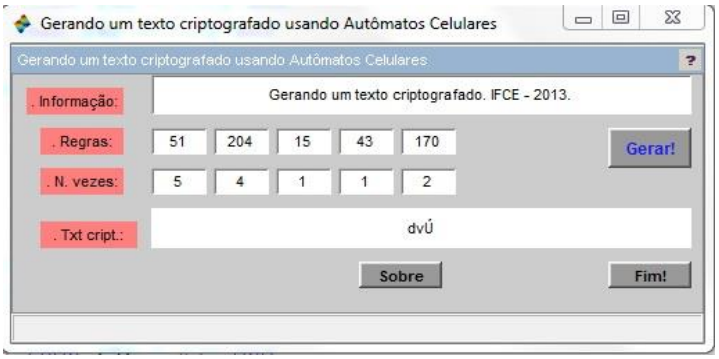


Fig. 7 – Codificação e o resultado logo após o processo de codificação, observar que foram gerados caracteres não imprimíveis.

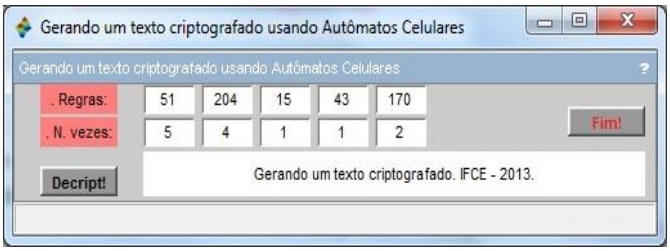


Fig. 8 – Aplicativo decriptador voltando para a mensagem original.

VII. CONCLUSÃO

Neste artigo foi realizado o estudo sobre uma técnica alternativa de criptografia que demonstrou ser possível a correção de erros, por meio da adição de bits de paridade e caracteres extras, gerados em mensagens, pela evolução de autômatos. Dois pares de regras (43,113 e 142,212) que propagavam erros na mensagem após a evolução do autômato foram tratados, possibilitando sua utilização direta por canais

de comunicação. Como discutido acima, nossa técnica ainda apresenta algumas limitações quanto ao uso de regras de evolução imperfeitas. Em trabalhos futuros pretende-se fazer uma criptoanálise aprofundada da técnica e expandir o código utilizado de tal forma que se possa trabalhar com padrões mais amplos e atuais como o Unicode UTF-8. Também se visa o melhoramento do software desenvolvido, ampliando suas funcionalidades e disponibilizando uma versão web do mesmo.

AGRADECIMENTOS

Os autores agradecem ao Instituto Federal de Educação, Ciência e Tecnologia do Ceará (IFCE), e ao Laboratório de Processamento Digital de Sinais (PDS/IFCE) pela infraestrutura e apoio cedidos para o desenvolvimento deste projeto.

REFERÊNCIAS

[1] OECD. Machine-to-Machine Communications: Connecting Billions of Devices. OECD Digital Economy Papers, No. 192, OECD Publishing. Disponível em: <http://dx.doi.org/10.1787/5k9gsh2gp043-en>. Acesso em 16 de agosto de 2013.

[2] SIMONITE, T. Math Advances Raise the Prospect of an Internet Security Crisis. MIT Technology Review, Cambridge, Massachusetts, Aug. 2013. Disponível em: <http://www.technologyreview.com/news/517781/math-advances-raise-the-prospect-of-an-internet-security-crisis/>. Acesso em 16 de agosto de 2013.

[3] UNO, D. N.; FALEIROS, A. C. Princípios de Criptografia Quântica. São José dos Campos, São Paulo, 2003.

[4] ALT, L. S.; FERREIRA, G. B.; MARTINS, M. V.; MARTINS, L. G. A.; OLIVEIRA, G. M. B. Um modelo criptográfico baseado em autômatos celulares com texto cifrado de tamanho variável. IX Encontro Interno & XIII Seminário de Iniciação Científica – PIBIC, 2009.

[5] TARDIVO FILHO, M.; HENRIQUES, M. A. A. Estudo sobre a Aplicação de Autômatos Celulares Caóticos em Criptografia. IV Encontro dos Alunos e Docentes do Departamento de Engenharia de Computação e Automação Industrial - EADCA, v. CD-ROM, pp. 1–4, Abr. 2011.

[6] BOJINOV, H. Neuroscience Meets Cryptography. 21st USENIX Security Symposium, Bellevue, Whashington, Aug. 2012.

[7] MACHICAO, J. M.; MARCO, A. G.; BRUNO, O. M. Chaotic Encryption Method Based on Life-Like Cellular Automata. arXiv [math.DS], Cornell University Library, Ithaca, New York, Dec. 2011.

[8] USP. Para pesquisadores do IFSC, criptografia baseada no caos é promessa de segurança online. Redação USP, Tecnologia, São Paulo, São Paulo, 7 Feb. 2012. Disponível em: <http://www5.usp.br/6242/para-pesquisadores-do-ifsc-criptografia-baseada-no-caos-e-promessa-de-seguranca-online/>. Acesso em 16 de agosto de 2013.

[9] CASTRO, M. L. A.; CASTRO, R. O. Autômatos Celulares: Implementações de Von Neumann, Conway e Wolfram. Revista de Ciência e Tecnologia, Vol. lll, Nº 3, 2008.

[10] FENWICK, J. W.; DOWELL, L. J. Electrical substation service-area estimation usingcellular automata: an initial report. InSAC ’99: Proceedings of the 1999 ACM symposiumon Applied computing, p. 560–565, 1999.

[11] REITE, C. A. A Local Cellular Model for Snow Crystal Growth. Chaos, Solitons & Fractals, Easton – PA, v.23, n. 4, p. 1111-1119, Feb. 2005.

[12] MELOTTI, G. Aplicação de Autômatos Celulares em Sistemas Complexos: Um estudo de Caso em Espalhamento de Epidemias. MACSIN-UFMG, Belo Horizonte,Fev.2009.

[13] MICHEL, G. V. Estudo de Mecanismo FEC para Transmissão Confiável em UDP. Porto Alegre, Jun. 2010.