UNIVERSIDADE DE BRASÍLIA
FACULDADE DE TECNOLOGIA
DEPARTAMENTO DE ENGENHARIA ELÉTRICA

EMPREGO DA ENGENHARIA REVERSA PARA CARACTERIZAÇÃO DO *MODUS OPERANDI* DAS MÁQUINAS CAÇA-NÍQUEIS QUANTO À PRÁTICA DE JOGO DE AZAR E OUTRAS FRAUDES

CLEVERSON ESTEVES DA SILVA

ORIENTADOR: RICARDO ZELENOVSKY
COORIENTADOR: GALILEU BATISTA DE SOUSA

DISSERTAÇÃO DE MESTRADO EM ENGENHARIA ELÉTRICA
ÁREA DE CONCENTRAÇÃO INFORMÁTICA FORENSE E SEGURANÇA DA INFORMAÇÃO

PUBLICAÇÃO: PPGENE.DM - 103 A/2012

BRASÍLIA / DF: 02/2012

UNIVERSIDADE DE BRASÍLIA
FACULDADE DE TECNOLOGIA
DEPARTAMENTO DE ENGENHARIA ELÉTRICA

EMPREGO DA ENGENHARIA REVERSA PARA CARACTERIZAÇÃO DO *MODUS OPERANDI* DAS MÁQUINAS CAÇA-NÍQUEIS QUANTO À PRÁTICA DE JOGO DE AZAR E OUTRAS FRAUDES

CLEVERSON ESTEVES DA SILVA

DISSERTAÇÃO DE MESTRADO SUBMETIDA AO DEPARTAMENTO DE ENGENHARIA ELÉTRICA DA FACULDADE DE TECNOLOGIA DA UNIVERSIDADE DE BRASÍLIA, COMO PARTE DOS REQUISITOS NECESSÁRIOS PARA A OBTENÇÃO DO GRAU DE MESTRE PROFISSIONAL EM INFORMÁTICA FORENSE E SEGURANÇA DA INFORMAÇÃO.

APROVADA POR:

RICARDO ZELENOVSKY, Doutor, ENE/FT
(ORIENTADOR)

HELVIO PEREIRA PEIXOTO, Doutor, ENE/FT
(EXAMINADOR INTERNO)

ANDREIA CRISTIANE STANGER, Doutora, Uninorte/AC
(EXAMINADOR EXTERNO)

ANDERSON CLAYTON ALVES NASCIMENTO, Doutor, ENE/FT
(SUPLENTE)

DATA: BRASÍLIA/DF, 29 DE FEVEREIRO DE 2012.

FICHA CATALOGRÁFICA

SILVA, CLEVERSON ESTEVES DA
Emprego da Engenharia Reversa para caracterização do *modus operandi* das máquinas caça-níquel quanto à prática de jogo de azar ou outras fraudes [Distrito Federal] 2012.
xiv, (112)p., 297 mm (ENE/FT/UnB, Mestre, Engenharia Elétrica, 2012).

Dissertação de Mestrado – Universidade de Brasília, Faculdade de Tecnologia. Departamento de Engenharia Elétrica.

1. Caça-níquel 2. Jogo de azar
3. Engenharia Reversa

I. ENE/FT/UnB. II. Título (Série)

REFERÊNCIA BIBLIOGRÁFICA

SILVA, C. E. (2012). Emprego da Engenharia Reversa para caracterização do *modus operandi* das máquinas caça-níquel quanto à prática de jogo de azar ou outras fraudes. Dissertação de Mestrado, Publicação PPGENE.DM - 103 A/2012, Departamento de Engenharia Elétrica, Universidade de Brasília, Brasília, DF, (112)p.

CESSÃO DE DIREITOS

NOME DO AUTOR: Cleverson Esteves da Silva
TÍTULO DA DISSERTAÇÃO: Emprego da Engenharia Reversa para caracterização do *modus operandi* das máquinas caça-níquel quanto à prática de jogo de azar ou outras fraudes.
GRAU/ANO: Mestre/2012.

Cleverson Esteves da Silva
Universidade de Brasília
Campus Universitário Darcy Ribeiro – Asa Norte
CEP 70910-900 – Brasília – DF - Brasil

À família, pilar de sustentação do caráter humano.

AGRADECIMENTOS

A Deus, que sempre esteve comigo, mesmo quando não estive com Ele.

Aos meus pais por não terem economizando esforços no sentido de proporcionar uma formação pautada nos princípios éticos e morais e também aos meus irmãos que sempre proporcionaram um ambiente harmônico de convívio, com cooperação e admiração mútua, tornando-nos incentivadores uns dos outros.

Ao Perito Criminal Federal Galileu Batista, que não só aceitou a função de coorientador, como também se empolgou com o aspecto pericial que o presente trabalho propôs-se a resolver, reduzindo a minha curva de aprendizado através do conhecimento transmitido.

Ao meu orientador Ricardo Zelenovsky que sempre se mostrou solícito, descartando toda e qualquer burocracia para que o trabalho fosse desempenhado conforme planejado, emprestando sua calma e paciência sempre que tudo parecia não ter solução e contribuindo para o meu desenvolvimento como pesquisador.

Aos meus amigos, extensão da minha família, que compreenderam a ausência e me motivaram a acreditar que os entraves surgidos durante a pesquisa não eram suficientemente grandes para inviabilizar a realização do trabalho. Em especial ao amigo Pablo Apolinário, que abriu as portas da sua casa para que eu pudesse me acomodar justamente no momento mais crucial: o período de adaptação à Capital Federal.

Aos colegas do Mestrado em Informática Forense, que além de estreitarem os laços entre as organizações policiais civis estaduais e policiais federais de várias regiões do País, proporcionaram uma generosa troca de conhecimento, em especial ao colega Sérgio Fava, que desde o primeiro momento colocou seu vasto acervo bibliográfico estrangeiro à disposição para que eu pudesse avaliá-los e decidir sobre quais deveria ou não adquirir.

Ao Perito Criminal Federal Hélvio Peixoto que, quando à frente da coordenação do curso, mostrou-se comprometido em garantir que todos os alunos do programa fossem atendidos em suas necessidades acadêmicas e logísticas da melhor forma possível.

A todos, os meus sinceros agradecimentos.

O presente trabalho foi realizado com o apoio do Instituto de Criminalística do Estado de Rondônia, com recursos do Programa Nacional de Segurança Pública com Cidadania – Pronasci, do Ministério da Justiça.

RESUMO

EMPREGO DA ENGENHARIA REVERSA PARA CARACTERIZAÇÃO DO MODUS OPERANDI DAS MÁQUINAS CAÇA-NÍQUEL QUANTO À PRÁTICA DE JOGO DE AZAR OU OUTRAS FRAUDES

Autor: Cleverson Esteves da Silva
Orientador: Ricardo Zelenovsky
Programa de Pós-graduação em Engenharia Elétrica
Brasília, fevereiro de 2012

Apesar de a prática de jogo de azar ser proibida no Brasil desde 1941, a popularização dos equipamentos de informática permitiu a criação de máquinas caça-níquel que, ao invés de se utilizarem de um conjunto de *hardware* específico, têm seu ambiente simulado por um programa de computador e são construídas com itens comuns de *hardware* já tidos como obsoletos. Com o objetivo de combater a utilização desses equipamentos, apreensões têm sido constantes em todo o território brasileiro, gerando um crescimento da demanda de exames periciais por parte dos órgãos oficiais de perícia criminal. Porém, nesses equipamentos, os detalhes do seu comportamento interno foram codificados pelo processo de compilação a que os programas foram submetidos, dificultando ao Perito Criminal o acesso às evidências da prática delituosa sem que seja empregada uma grande quantidade de horas de análise em cada equipamento apreendido. Diante da incerteza do comportamento interno dos programas que gerenciam tais equipamentos, os trabalhos periciais não têm sido conclusivos a ponto de esclarecer se estes oferecem ou não propriedades que possibilitem ao apostador influenciar no resultado final. Utilizando-se de técnicas de Engenharia Reversa, o presente trabalho objetiva apresentar um conjunto de elementos materiais que evidencie a prática de jogo de azar ou de fraudes diversas nos aplicativos das máquinas tipo "Halloween" analisados. Em um segundo momento, são demonstradas as similaridades existentes entre jogos aparentemente distintos, permitindo que versões do aplicativo não abrangidas pelo estudo sejam inferidas quanto à semelhança de comportamento. Ainda, objetivando instruir futuras análises de aplicativos que não pertençam à família "Halloween", é exibida a metodologia empregada para facilitar a obtenção das informações necessárias à atividade pericial e que poderá ser adaptada a outras situações que envolvam aplicativos da mesma natureza.

ABSTRACT

THE USE OF REVERSE ENGINEERING TO DEFINE THE *MODUS OPERANDI* OF SLOT MACHINES AS TO PRACTICE OF GAMBLING OR OTHER FRAUDS

Author: Cleverson Esteves da Silva
Supervisor: Ricardo Zelenovsky
Programa de Pós-graduação em Engenharia Elétrica
Brasília, february of 2012

Despite gambling be forbidden in Brazil since 1941, the growing access to computer equipment brings a new kind of slot machines that, instead of using a specific hardware group, has its environment simulated by a computer program and is built with common and obsolete hardware items. In order to combat the use of such equipment, seizures have been constant throughout the Brazilian territory, bringing a growing of forensic examination demand for the criminal institutes. However, the details of its internal behavior are hidden by the compilation process, making it difficult to access to the crime evidences, demanding lots of hours in the analysis on such equipments. Due to the uncertainty about internal behavior of the management programs that controls these equipments, the forensic exams has not been as conclusive whether they offer properties to allow player influence on final result. Using reverse engineering techniques, this study presents a set of evidences about gambling and frauds in the "Halloween" machines applications. In a second step, are shown the similarities between apparently different games, allowing inferring conclusions on about applications not covered by this study. Furthermore, in order to instruct future analyses on machines of different class, here is shown the methodology used to obtaining the necessary information by forensic exam, that could be adapted to other situations involving same kind of applications not covered in this study.

SUMÁRIO

1. INTRODUÇÃO 15

1.1. PROBLEMÁTICA 15

1.2. ESCOPO E OBJETIVOS DA PESQUISA 16

1.3. METODOLOGIA 17

1.4. RESULTADOS ESPERADOS 18

1.5. ORGANIZAÇÃO DO TRABALHO 20

2. INFORMÁTICA FORENSE 21

2.1. NATUREZA CIENTÍFICA DA FUNÇÃO PERICIAL 24

3. JOGOS DE AZAR 26

3.1. MÁQUINAS CAÇA-NÍQUEL (*SLOT MACHINES*) 27

3.2. LEGALIDADE DOS JOGOS DE AZAR 31

3.3. DEMANDA EM TERRITÓRIO BRASILEIRO 32

4. ENGENHARIA REVERSA DE SOFTWARE 34

4.1. ÁREAS DE APLICAÇÃO 35

4.2. ABORDAGENS 36

4.3. PROTEÇÕES CONTRA ENGENHARIA REVERSA 37

4.4. FERRAMENTAS 38

4.5. ARQUITETURAS 39

5. TRABALHOS CORRELATOS 43

5.1. ANÁLISE ESTATÍSTICA DE MÁQUINAS DE VÍDEO BINGO 43

5.2. ENGENHARIA REVERSA DE MALWARE 45

5.3. ANÁLISE DO ISOMORFISMO DE OBJETOS EXECUTÁVEIS 47

6. ANÁLISE DE MÁQUINAS CAÇA-NÍQUEL 50

6.1. USABILIDADE DO SOFTWARE 53

6.2. ITENS DE HARDWARE 59

6.2.1. Dispositivo de Armazenamento 59

6.2.2. Conjunto Microprocessador 60

6.2.3. Teclado **60**
6.2.4. Noteiro **62**
6.3. AMBIENTE OPERACIONAL DO JOGO **63**
6.3.1. Identificação dos arquivos **63**
6.3.2. Arquivos Protegidos por Senha **72**
6.3.3. Arquivos Dissimulados **74**
6.4. ESTRUTURA INTERNA DO JOGO **75**
6.4.1. Tela de Seleção de Jogos **75**
6.4.2. Aplicativo Principal do Jogo **76**
6.4.2.1. Ambiente de Configuração 76
6.4.2.2. Primeiro Nível de Segurança 77
6.4.2.3. Segundo Nível de Segurança 80
6.4.2.4. Terceiro Nível de Segurança 81
6.4.2.5. Senhas de Acesso dos Níveis de Segurança 81
6.4.3. Base de Dados Principal **83**
6.4.3.1. Decodificação de Informações da Base 83
6.4.3.2. Parâmetros de Configuração Fraudulentos 85

7. SIMILARIDADE ENTRE JOGOS **89**
7.1. COMPARAÇÃO POR VALORES DE HASH **93**
7.2. COMPARAÇÃO BLOCO A BLOCO DE EXECUTÁVEIS **95**
7.3. COMPARAÇÃO ISOMÓRFICA DE GRAFOS **96**

8. CONCLUSÕES **102**
8.1. DIFICULDADES ENCONTRADAS **103**
8.2. TRABALHOS FUTUROS **104**

REFERÊNCIAS BIBLIOGRÁFICAS **105**

A – LIGAÇÃO DOS EMULADORES DE TECLADO **111**

LISTA DE TABELAS

Tabela 5.1. Estatísticas de Jogadas. 44

Tabela 6.1. Relação de jogos oferecidos pelos equipamentos analisados..... 52

Tabela 6.2. Correspondência dos pinos da porta paralela com os valores ASCII gerados 62

Tabela 6.3. Conteúdo comum entre os dispositivos de armazenamento das máquinas..... 64

Tabela 6.4. Conteúdo do diretório “data” comum entre as máquinas..... 65

Tabela 6.5. Conteúdo do diretório “dos” comum entre as máquinas..... 66

Tabela 6.6. Arquivos com extensão “sys” presentes na maquina ilustrada na Figura 6.4..... 70

Tabela 6.7. Comparativo de conteúdo dos arquivos com extensão “sys”..... 71

Tabela 6.8. Conteúdo do arquivo “jogos.bin” da máquina ilustrada na Figura 6.2 72

Tabela 6.9. Tradução do conteúdo do arquivo “jogos.bin” da máquina da Figura 6.4..... 74

Tabela 6.10. Relação de senhas encontradas nos 23 jogos examinados 83

Tabela 6.11. Dicionário de dados da tabela “base.dbf” 84

Tabela 7.1. Comparativo entre os 23 jogos examinados..... 90

Tabela 7.2. Valores de hash gerados com base nos arquivos binários..... 94

Tabela 7.3. Relacionamento das diferenças em byte de cada grupo da Tabela 7.2 95

Tabela 7.4. Quantidade de funções identificadas nos arquivos descompactados 98

LISTA DE FIGURAS

Figura 3.1. Componentes de uma máquina caça-níquel mecânica 27

Figura 3.2. Sistema de acionamento das máquinas caça-níquel controladas por computador 28

Figura 3.3. Semelhança externa entre as máquinas caça-níquel e as de fliperama. 30

Figura 4.1. Endereçamento de registradores em arquiteturas de 32 bits 41

Figura 5.1. Exemplo de grafo de controle de fluxo da função “sub_401260” 48

Figura 6.1. Evolução das máquinas caça-níquel quanto ao porte 51

Figura 6.2. Tela para escolha de jogos oferecidos pela Máquina 8 da Tabela 6.1 54

Figura 6.3. Tela principal do jogo “Halloween I” com a opção “linha 9” ativada 55

Figura 6.4. Tabela de combinações de premiação do jogo “Halloween I” 67

Figura 6.5. Tela de bonificação 58

Figura 6.6. Composição gráfica ilustrando o interior dos equipamentos 59

Figura 6.7. Diferenças entre as máquinas caça-níquel com jogo gravado em placa de circuito impresso ou em memória semicondutora 61

Figura 6.8. Modelo de noteiro P70 63

Figura 6.9. Tela de administração do sistema 76

Figura 6.10. Tela de leitura parcial 77

Figura 6.11. Tela de leitura oficial acessível através de senha oculta 78

Figura 6.12. Tela dos últimos pagamentos efetuados 79

Figura 6.13. Tela das cinco últimas entradas de crédito 79

Figura 6.14. Tela de controle de acesso por senha de segurança 80

Figura 6.15. Tela de configuração de parâmetros acessível por senha de segurança 80

Figura 6.16. Localização da senha de acesso à tela de configuração de parâmetros 81

Figura 6.17. Localização da senha de acesso à tela de leitura oficial 82

Figura 6.18. Localização da senha de acesso à tela “Config. Impressora 82

Figura 7.1. Arquivo “tela1.dat” presente no arquivo compactado “bigb1ac.sys” 91

Figura 7.2. Arquivo “tela1.dat” presente no arquivo compactado “buca2.sys” 91

Figura 7.3. Arquivo “tela1.dat” presente no arquivo compactado “hallofor.sys” 92

Figura 7.4. Arquivo “tela.dat” presente no arquivo compactado “oro2ac.sys” 93

Figura 7.5. Identificação do segmento de código na ferramenta IDA Pro 96

Figura 7.6. Grafo das funções encontradas no arquivo binário do pacote “bigb1ac.sys”..... 97

Figura 7.7. Grafo dirigido do arquivo binário do pacote “bigb1ac.sys” depois de descompactado 98

Figura 7.8. Grafo dirigido das funções do arquivo binário do pacote “hallofor.sys”..... 99

Figura 7.9. Grafo de controle de fluxo da função “sub_32028” do arquivo binário referente ao grafo dirigido da Figura 7.7..... 99

Figura 7.10. Grafo de controle de fluxo da função “sub_31FAC” do arquivo binário referente ao grafo dirigido da Figura 7.8..... 100

Figura A.1. Esquema de conexão do emulador de teclado HW002 com 10 botões..... 111

Figura A.2. Esquema de conexão do emulador de teclado AVD005 com 8 botões..... 112

LISTA DE SÍMBOLOS, NOMENCLATURA E ABREVIAÇÕES

AMD - *Advanced Micro Devices*, fabricante de processadores.

API - *Application Programming Interface*, conjunto de funcionalidades de um *software*.

ASCII - *American Standard Code for Information Interchange,* conjunto de caracteres.

CP - Código Penal.

CPP - Código de Processo Penal.

CRT - *Cathode Ray Tube.*

DB25 - Conector de porta paralela com 25 pinos.

DBF - *Database File*, extensão de arquivos da base de dados dBase.

DIN - Deutsches Institut für Normung, Interface de comunicação do teclado.

DOS - *Disk Operating System*, Sistema operacional *stand alone*.

EAX - *Extended Accumulator*, registrador 32-bits.

EBP - *Extended Stack Base Pointer*, registrador 32-bits.

EBX - *Extended Base Index*, registrador 32-bits.

ECX - *Extended Counter*, registrador 32-bits.

EDI - *Extended Destination Index*, registrador 32-bits.

EDX - *Extended Data*, registrador 32-bits.

ESI - *Extended Source Index*, registrador 32-bits.

ESP - *Extended Stack Pointer*, registrador 32-bits.

FAT - *File Allocation Table*, sistema de arquivo.

FTK - *Forensic Toolkit*

ISA - *Industry Standard Architecture*, barramento de comunicação.

LCD - *Liquid Crystal Display*.

MD5 - *Message Digest 5*

NIST - *National Institute of Standards and Technology*.

OOV - *Order of Volatility*, ordem de volatilidade das informações digitais armazenadas.

PC - *Personal Computer*, arquitetura de computadores.

PCI - *Peripheral Component Interconnect*, barramento de comunicação.

PS/2 - *Personal System/2*, interface de comunicação.

RAM - *Random Access Memory*, memória principal.

RAW - Formato de imagem utilizado para espelhamento

SATA - *Serial Advanced Technology Attachment*, barramento de comunicação.

SPAM - *Spiced Ham*, mensagens eletrônicas não solicitadas enviadas em massa.

UP - *Unified Process*, Metodologia de desenvolvimento ágil.

USB - *Universal Serial Bus*, interface de comunicação.

ZIP - Formato de compressão com perda.

1. INTRODUÇÃO

Assim como a tecnologia tem invadido o cotidiano dos cidadãos de conduta socialmente aceitável, de forma acelerada e irreversível, trazendo comodidade e celeridade para tarefas anteriormente dispendiosas, a criminalidade tem se especializado e utilizado a tecnologia a seu favor, quer seja aprimorando velhas práticas delituosas, quer seja criando novas modalidades de lesar particulares, empresas, instituições públicas, ou qualquer outro que esteja ao seu alcance (Costa, 2003).

Embora não se tenha ainda uma legislação específica acerca dos tipos penais cometidos com o auxílio de equipamentos de informática, como afirmam Rodrigues (1999) e Nogueira (2002), alguns crimes podem ser analisados por analogia com base na Legislação Penal vigente como, por exemplo, os crimes de estelionato, danos, pedofilia, dentre inúmeros outros, sendo tal preceito extensivo ao Código de Processo Penal, que versa a respeito, dentre outros aspectos da processualística penal, da função do Perito Criminal, agente do Estado responsável pela realização dos mais diversos tipos de exames sempre que uma infração penal deixa vestígios, não importando se esses são físicos ou lógicos.

1.1. PROBLEMÁTICA

Algumas modalidades de crime praticadas através de equipamentos eletrônicos encontram uma barreira na análise do seu *modus operandi*[1], pois as instruções lógicas responsáveis por gerenciar tais recursos apresentam-se codificadas em linguagem de máquina. Um exemplo notório de tal prática é a máquina caça-níquel simulada por programa computacional, conhecida também como videobingo, cujas constantes apreensões vêm sendo amplamente noticiadas nos mais diversos meios de comunicação em todo o território nacional.

Em equipamentos dessa natureza, a caracterização ou não de comportamento que configure a prática de jogo de azar, previsto no Art. 50 da Lei de Contravenções Penais, depende única e exclusivamente da lógica de funcionamento do *software* aplicativo carregado na memória de acesso aleatório (RAM) durante a inicialização do equipamento a partir do seu dispositivo de armazenamento, pois não há como garantir que o processamento interno da interação seja efetivamente realizado da forma como se pressupõe através dos elementos

[1] Termo em latim usado no ambiente forense que significa "modo de operação".

da interface.

Diferentemente das máquinas caça-níquel convencionais, cujo sistema é acionado por uma única alavanca que dispara a movimentação dos discos, as máquinas de videobingo possuem diversos botões com funcionalidades distintas que sugerem a possibilidade de serem utilizados de acordo com a habilidade do apostador e assim maximizar ou minimizar as suas chances de ganho ou perda. Essa incerteza do funcionamento interno dos aplicativos de tais equipamentos tem sido o principal argumento dos advogados de defesa para impedir o enquadramento no dispositivo legal que trata dos jogos de azar, alegando que não se trata de sorte, mas sim de técnica.

A argumentação de defesa ganha maior amparo com a semelhança guardada por tais equipamentos com os jogos de fliperama existentes há décadas, onde o jogador interage através de diversos botões e pode influenciar no resultado final a cada ficha inserida, já que tende a adquirir experiência com o comportamento do jogo diante de determinada ação.

1.2. ESCOPO E OBJETIVOS DA PESQUISA

É senso comum de que a definição de uma metodologia de aferição de resultados para uma determinada tecnologia está diretamente ligada à necessidade de um amplo conhecimento das suas características, uma vez que se busca verificar se os resultados apresentados diante de uma determinada interação condizem com o comportamento que se espera dela.

Diante da carência de trabalhos que analisem em profundidade o comportamento interno dos programas computacionais que simulam o ambiente de uma máquina caça-níquel, bem como da escassez de documentação técnica sobre os elementos de *software* ou sobre os itens específicos de *hardware*, supostamente atribuídos a suspeita de ilegalidade que paira sobre esses equipamentos, o presente estudo tem por objetivo a definição de um conjunto de procedimentos que permita traçar um perfil de comportamento de tais máquinas com base no modo de operação inferido a partir dos elementos materiais coletados.

Através do conhecimento extraído a partir da análise do comportamento comum apresentado por esses equipamentos, objetivou-se fornecer um conjunto de elementos materiais concretos que venham a subsidiar os exames realizados por Peritos Criminais de todos os Estados da Federação, a fim de que eles possam ser conclusivos em seus laudos no que se refere à prática de jogo de azar e outras fraudes, encerrando definitivamente a discussão que vem se prolongando no âmbito forense desde o surgimento de tais máquinas.

Depois de caracterizado o modo de operação das máquinas, é possível ainda, com base nos resultados alcançados, instruir pesquisas futuras que versem sobre o desenvolvimento de uma metodologia de aferição de resultados ou ainda de uma ferramenta que possa realizar a análise automatizada do comportamento de aplicativos semelhantes.

Dos equipamentos recebidos a partir do ano de 2009 na Seção de Criminalística de Ji-Paraná (RO) para análise, somente integram o objeto de estudo do presente trabalho as máquinas que apresentam cartão de memória semicondutora conectada ao *slot* IDE da placa-mãe e que se mostraram operantes à época do recebimento. Aquelas máquinas, cuja aplicação está gravada em uma placa de circuito impresso que substitui a placa-mãe ou em placas que são inseridas em *slots* PCI ou ISA, foram descartadas da amostra por corresponderem a uma geração de equipamentos que está cada dia mais obsoleta devido à especificidade do *hardware*.

Assim, o objeto de estudo do presente trabalho consiste de 08 (oito) exemplares de máquinas caça-níquel denominadas "Halloween", que oferecem ao todo 18 (dezoito) opções de jogos aparentemente distintos, mas que apresentam jogabilidade semelhante.

Embora os exames tenham se restringido aos equipamentos disponíveis na Seção de Criminalística de Ji-Paraná, observa-se que a aparência deles assemelha-se à dos aparelhos examinados pelo Instituto de Criminalística do Estado de Rondônia e por órgãos periciais de outras regiões brasileiras, já que apresentam a mesma forma externa de interação e a mesma forma interna de exibição de resultados.

1.3. METODOLOGIA

Por apresentar uma dezena de instrumentos de interação e diversos elementos de interface, que elevariam exponencialmente a quantia de combinações necessárias para se traçar um perfil confiável de comportamento desses equipamentos, através da realização de uma análise estatística baseada em exaustivas execuções, a técnica empregada por Nogueira (2002) foi desconsiderada.

Assim, em virtude do caráter de ilegalidade que envolve tanto as aplicações que simulam o ambiente de uma máquina caça-níquel, como aquelas dotadas de código malicioso, sendo ambas concebidas e distribuídas sem qualquer documentação que especifique o seu comportamento interno, adotou-se a metodologia de análise definida por Vênere (2009, p.118-143), que emprega técnicas de Engenharia Reversa para extrair informações sobre o

comportamento de aplicações, partindo daquelas obtidas mais facilmente através de funcionalidades de ferramentas específicas, para aquelas mais complexas, que necessitam da análise de código *assembly* (Casey, Malin & Aquilina, 2008).

Diante da ininteligibilidade do código executável em representação binária, a análise das funcionalidades de tais aplicativos sem o acesso direto ao código-fonte somente se dará através de técnicas de Engenharia Reversa, pois conforme defende Pressman (2006): "a engenharia reversa de *software* é o processo de análise de um programa, em um esforço para representá-lo em uma abstração mais alta do que o código-fonte. (…) As ferramentas de engenharia reversa extraem informação do projeto de dados, arquitetural e procedimental, para um programa existente".

1.4. RESULTADOS ESPERADOS

A partir do objetivo do estudo proposto espera-se a caracterização do comportamento geral da aplicação desde o momento em que o apostador aciona o botão que dispara o mecanismo de sorteio, até o momento em que o resultado é exibido na tela.

Embora, através da Engenharia Reserva, seja possível gerar um modelo completo do sistema analisado, alguns detalhes somente têm utilidade quando há necessidade de reconstrução do aplicativo. No caso em lide, a compreensão parcial tem a capacidade de resultar na informação que se deseja extrair da aplicação.

Assim, pretende-se num primeiro momento elucidar não só a jogabilidade (interação homem-computador) apresentada pela interface, como também o comportamento interno da aplicação em relação ao ambiente operacional do jogo. Através da adaptação da metodologia de análise de código malicioso apresentada por Vênere (2009, p.118-143), espera-se produzir um modelo comportamental do sistema que permita garantir ou descartar, de forma cabal, a prática de jogo de azar ou outras fraudes nesse tipo de equipamento.

As máquinas caça-níquel em questão comumente apresentam mais de um jogo em seu dispositivo de armazenamento permanente, cuja escolha é realizada através de uma tela preliminar exibida durante a inicialização dos equipamentos. Uma procura pelo termo caça-níquel nos principais motores de busca retornará algumas centenas de jogos destinados a essas máquinas com comportamento aparentemente idêntico.

Uma vez concebido um modelo comportamental de um desses jogos, é necessário que

sejam elencados os pontos de similaridade existentes quanto ao comportamento de jogos aparentemente distintos, possibilitando que versões do aplicativo principal de máquinas tipo "Halloween" não analisadas neste trabalho sejam inferidas quanto à prática da modalidade de jogo de azar ou fraudes diversas, tanto através da comparação direta do arquivo executável, como através da identificação das características operacionais e comportamentais apresentadas pelo presente estudo.

Embora o presente trabalho tenha como objetivo principal a evolução do estado da arte, no que se refere ao comportamento interno das máquinas caça-níquel, não se pretende apenas identificar e caracterizar tais equipamentos, mas também estabelecer um grau de similaridade entre jogos distintos que permita concluir sobre as práticas de jogo de azar ou outras fraudes.

Pretende-se também apresentar uma metodologia de análise que auxilie o exame pericial de equipamentos similares apreendidos em todos os Estados da Federação, fornecendo diretrizes que visam à obtenção de informações relevantes à atividade pericial, inclusive em aplicações da mesma natureza que não pertençam à família "Halloween".

Com isso, objetiva-se reduzir a necessidade de dispensar excessivas horas de análise na compreensão de algoritmos de codificação de dados ou no correlacionamento de arquivos, assim como estabelecer um padrão de análise desse tipo de equipamento.

1.5. CONVENÇÕES

Por convenção, os termos "packing" e "empacotamento" são utilizados para se referir ao método de compactação de um arquivo binário como forma de ofuscamento da linguagem de máquina ou de redução de espaço de armazenamento, sem comprometer a sua propriedade de ser carregado na memória e executado diretamente. Por sua vez, o termo "compressão" é utilizado quando um determinado arquivo (ou conjunto deles), binário ou não, é submetido a um processo de redução de espaço físico (compactação), com ou sem a adoção de uma senha de descompactação e que necessite de uma ferramenta de terceiros para voltar ao estado original.

Sempre que o termo "compactação" for utilizado, referir-se-á à ação de redução do tamanho físico de um determinado arquivo de forma a abranger tanto as técnicas de empacotamento, como as de compressão.

1.6. ORGANIZAÇÃO DO TRABALHO

O Capítulo 2 contextualiza a informática forense em relação aos conceitos empregados pela ciência Criminalística quanto a sua peculiaridade no que se refere à identificação, preservação coleta e tratamento das evidências digitais.

No Capítulo 3 é abordada a discussão sobre a definição de jogo de azar dada pela legislação vigente, correlacionando tal definição com o conceito de probabilidade matemática e as características comumente apresentadas por máquinas caça-níquel.

O Capítulo 4 encarrega-se do embasamento teórico referente às modalidades de análise "*off-line*" e análise "*online*" da Engenharia Reversa, apresentando desde o aparato ferramental utilizado até as diferenças existentes quanto à arquitetura.

O Capítulo 5 apresenta os trabalhos correlatos que foram utilizados para instruir a pesquisa e alcançar os resultados pretendidos.

O Capítulo 6 encarrega-se de explicar a metodologia de análise utilizada e expor detalhadamente os experimentos realizados com as máquinas caça-níquel que integram o objeto de estudo do presente trabalho, no que se refere às peculiaridades do *hardware*, usabilidade do *software*, ambiente operacional, programas de apoio utilizados pela máquina e comportamento interno da aplicação principal.

Ao Capítulo 7 cabem as abordagens utilizadas para determinar a existência de similaridades entre aplicações aparentemente distintas.

Por fim, o Capítulo 8 apresenta as conclusões alcançadas a partir dos experimentos realizados conforme descrito no capítulo anterior, apresentando sugestões de trabalhos futuros com base no conhecimento obtido acerca do presente trabalho.

2. INFORMÁTICA FORENSE

Em detrimento de outras ciências que vêm evoluindo ao longo de séculos, a Informática engloba um conjunto de ciências da informação recentes e não completamente consolidadas, que na última década passou por uma normatização no Brasil, exigindo a intervenção direta do Ministério da Educação e Cultura no sentido de categorizar os diversos cursos de graduação que surgiam em todo o território nacional (SBC, 2011a e SBC, 2011b).

A Informática encarrega-se do tratamento automatizado e racional da informação com o emprego do computador eletrônico na ciência da informação (Michaelis, 2011a). Pode ser considerada ainda como o "conjunto de aplicações dessa ciência, através da utilização de máquinas (computadores) e programas (*software*)" (Aurélio, 2011). Diante dessas definições, observa-se que o conteúdo abrangido pela Informática é mais amplo do que pode aparentar, não se limitando aos computadores pessoais. Ao contrário disso, estende-se a todo e qualquer equipamento eletrônico capaz de realizar computações (cálculos) de forma automatizada.

Neste capítulo são apresentados conceitos inerentes à função pericial quanto ao seu amparo legal e a sua natureza científica, destacando-se as peculiaridades referentes aos crimes de informática em relação aos crimes ditos convencionais.

Diante da necessidade de investigar ações delituosas praticadas através de equipamentos eletrônicos, utilizando-se de métodos cientificamente comprovados para obtenção da materialidade necessária a sua comprovação, surgiu em meio à ciência Criminalística o conceito de Informática Forense. Cabe salientar que os dicionários da língua pátria tratam o termo "forense" como aquilo que é relativo aos tribunais, ou seja, de interesse do Poder Judiciário. Embora haja divergência quanto à nomenclatura dada ao emprego de áreas da Informática para esclarecimento do *modus operandi* de ações que utilizem equipamentos eletrônicos como meio ou fim de práticas delituosas, adotando termos como computação forense, criminalística computacional, forense digital, perícia eletrônica, dentre outros, há um consenso entre os autores (Freitas, 2006, p.1; Espindula, 2006, p.363; Costa, 2003, p.7 e Espindula & Sousa, 2011, p.300) sobre a finalidade dessa área do conhecimento.

Com base na comparação realizada por Espindula & Sousa (2011, p.300) entre os crimes praticados no "mundo real" e no "mundo virtual", é possível inferir que os fatos estudados

pela Informática Forense devem convergir com os princípios básicos da investigação pericial (determinação de autoria e obtenção da materialidade) com base nas evidências processadas através dos vestígios coletados no ambiente analisado.

Na matéria penal, a determinação da autoria consiste na identificação do agente que praticou a ação delituosa e que consequentemente receberá as sanções previstas na legislação vigente, enquanto que a obtenção da materialidade tem como objetivo atender à exigência descrita pelo art 158 do CPP (Espindula, 2006, p.13): "quando a infração deixar vestígios, será indispensável o exame de corpo de delito, direto ou indireto, não podendo supri-lo a confissão do acusado.".

Tal exigência deve-se ao poder discricionário do poder judiciário, atribuído pelo art. 157 do CPP, ao facultar ao magistrado a possibilidade de formação da sua convicção pela livre apreciação da prova, quer seja ela testemunhal ou material, não ficando ainda adstrito ao laudo, "(…) podendo aceitá-lo ou rejeitá-lo, no todo ou em parte", conforme exposto em seu art. 182, (Dorea, Stumvoll & Quintela, 2003, p.3).

Embora aquele dispositivo legal seja enfático ao determinar a realização de exame pericial de todo o ambiente utilizado para a prática delituosa (corpo de delito), mesmo que haja uma confissão formal, a comprovação do fato em estudo deve estar pautada em um conjunto de elementos relacionados entre si, que possam sem confirmados através do emprego de métodos cientificamente amplamente difundidos (Espindula, 2006, p.79).

Para Espindula & Sousa (2011, p.300) e Freitas (2006, p.1), a Informática Forense emprega técnicas e métodos científicos para localizar, preservar, analisar e apresentar elementos materiais de práticas delituosas localizados em dispositivos computacionais dos mais diversos tipos, que tenham sido utilizados para a prática de crimes ou ainda, que tenham sido alvo delas.

Cabe salientar a diferença existente entre o termo "vestígio", descrito na norma processual, e o termo "evidência", citado por Freitas (2006, p.1). Embora ambos sejam comumente utilizados para descrever o elemento material que possui relação com o fato delituoso em estudo, seja em ambiente real ou virtual, o "vestígio" refere-se a todo elemento material que se presume ter ligação com fato (Espindula, 2006, p.85). Com isso, restringe-se somente aos dois primeiros procedimentos da informática forense (localização e preservação).

Nesta etapa inicial da informática forense, devem ser identificados (descritos e

categorizados) e preservados (coletados de forma a garantir a sua idoneidade) todos os elementos materiais encontrados, sem que necessariamente haja relacionamento entre eles (Dorea, Stumvoll & Quintela, 2003, p.67-73).

Após a etapa de análise e processamento dos vestígios, são separados aqueles que correspondem efetivamente à comprovação da infração penal que se pretende apurar (evidência), deixando de ser mero indicativo (vestígio) e passando a configurar o *modus operandi* da ação, correlacionando-se entre si através da linha do tempo que determina a dinâmica da ação do infrator desde o seu início até o término (Espindula, 2006, p.86).

A garantia de punibilidade de um determinado fato delituoso está diretamente relacionada à competência do Perito em atender com sucesso tanto os seus três primeiros procedimentos básicos (localização, preservação e análise), que são de cunho técnico-científico, quanto o último deles (apresentação), que é eminentemente didático. Apresentar de forma clara os resultados obtidos dos três procedimentos anteriores, demonstrando os métodos científicos que levaram a eles, é tão importante quanto possuir o domínio do conhecimento empregado. Somente assim o resultado poderá ser repetido posteriormente caso haja necessidade de uma contraprova.

Não bastasse isso, na Informática Forense, há ainda o fator de que "em muitos casos, as únicas evidências disponíveis são as existentes em formato digital" (Freitas, 2006, p.2).

Como diferencial em relação às demais áreas da Criminalística, a Informática Forense possui duas formas de análise e cada uma delas resulta em um conjunto distinto de vestígios. As formas de coleta são definidas por Costa (2003, p.9) como "análise *online*" e "análise *off-line*", estando a primeira delas incumbida da extração de informações voláteis (existentes somente enquanto o equipamento encontra-se em operação), ao passo que a segunda encarrega-se dos demais vestígios registrados de forma permanente nos dispositivos de armazenamento existentes.

A sequência de coleta de tais vestígios deve ser procedida em conformidade com a ordem de volatilidade (*Order of Volatility* - OOV) das informações nos dispositivos de armazenamento, partindo do dispositivo mais volátil (análise ao vivo) para o mais permanente (análise *off-line*) (Farmer & Venema, 2007, p.6).

Outra peculiaridade da Informática Forense é que diferentes crimes resultam em um conjunto também diferente de vestígios, implicando na adoção de técnicas e no emprego de ferramentas distintas para cada fato que se deseja esclarecer. Em alguns desses crimes (ou

contravenções penais), como, por exemplo, a prática de jogo de azar através de máquinas caça-níquel simuladas por programas de computador, a informação que se deseja obter não está disponível em texto claro, mas sim em uma combinação de símbolos aparentemente ininteligível para o ser humano.

Desta forma, em dadas circunstâncias, "para compreender a funcionalidade de um artefato, um equipamento eletrônico ou um conjunto de programas de computador, deve-se proceder à realização da engenharia reversa" (Tocchetto & Spindula, 2005, p.213).

Porém, ao utilizar-se da Engenharia Reversa para entender o comportamento de um sistema de informação qualquer, Tocchetto & Spindula (2005, p.213) defendem que não há necessidade de ser analisada a totalidade de suas funções. Basta que sejam examinadas aquelas funcionalidades que sejam suficientes para comprovar o ato delituoso que se está apurando. As funcionalidades examinadas devem expor de forma clara como determinada informação de entrada é transformada na informação de saída exibida, seja ela textual, gráfica ou comportamental.

2.1. NATUREZA CIENTÍFICA DA FUNÇÃO PERICIAL

Ao confrontar as demais diretrizes do Código de Processo Penal, quanto à função pericial do Estado, com o princípio constitucional da não obrigatoriedade de um indivíduo de produzir provas contra si mesmo, largamente disseminado com base na interpretação da Carta Magna, verifica-se que apesar de não haver hierarquia entre as provas de um processo (material e testemunhal), a prova pericial (material) é a única que tem por obrigação ser imparcial.

Essa imparcialidade deve-se ao fato de a prova pericial ser produzida por Perito Oficial, investido em cargo público (Dorea, Stumvoll & Quintela, 2003, p.78) ou nomeado pela autoridade judicial (*ad hoc*[2]), que está sujeito a incorrer no crime de Falsa Perícia, tipificado pelo art. 342 do CP, caso não sejam observados os requisitos técnico-científicos do exame realizado, que venham a produzir resultado inverídico ou incorreto de forma intencional.

Com base na afirmação de Reis (2006, p.44), que define a perícia Criminalística como uma pesquisa "científica por excelência", é possível inferir que a Informática Forense também o

[2] Termo utilizado para definir um profissional que é nomeado para esclarecer um determinado ato.

é, uma vez que, por ser uma das áreas de atuação da Criminalística, está amparada pela mesma ideologia, não sendo facultada ao Perito a possibilidade de concluir seus exames através de elementos que não possam ser comprovados posteriormente.

O Perito Criminal tem como objetivo único e exclusivo a busca da verdade sobre um determinado fato, através do estabelecimento não só da sua causa, como também da confirmação (ou negação) de autoria. Para tanto, deve utilizar-se de todo e qualquer aparato tecnológico disponível.

O autor (Reis, 2006, p.44) frisa ainda que um Perito "não pode ser displicente ou tendencioso diante de verdades que estão latentes", devendo a margem de erro (do método empregado) ser a menor possível, já que o trabalho realizado por um Perito Criminal poderá servir de fonte de pesquisa para exames futuros e assim trazer evolução não só para o fato delituoso em estudo, como também para o estado da arte de determinada área da ciência Criminalística.

Neste capítulo foi possível verificar que alguns crimes de informática necessitam de técnicas avançadas de análise de arquivos binários para se ter acesso às características internas do seu comportamento, mesmo que parcial, já que para uma determinada entrada é possível obter a mesma saída de diversas formas diferentes.

3. JOGOS DE AZAR

Apesar de a informática forense ser encarregada de uma grande quantidade de crimes praticados tendo equipamentos eletrônicos como meio ou fim, este capítulo trata de delimitar o escopo deste trabalho às máquinas caça-níquel. Para tanto, realiza uma breve discussão acerca da distinção entre jogo de azar e fraude, sob o aspecto probabilístico, situando tais equipamentos e cada uma das modalidades lesivas conforme o comportamento apresentado. Para comprovar que não se trata de um problema pontual de uma determinada classe social ou região do País, são apresentados ainda casos de apreensões em vários Estados brasileiros.

O jogo de azar é "aquele cujo resultado não depende da habilidade ou do cálculo, mas exclusivamente da sorte" (Michaelis, 2011b), sendo esta a definição utilizada pela Lei de Contravenções Penais para tipificar a conduta.

Bello (2008, p.53), por sua vez, define jogo de azar como sendo aquele onde a única ação disponível ao jogador é aguardar o resultado processado pelo mecanismo do jogo, quer ele seja manual ou automatizado, na esperança de que seja compatível com o valor escolhido, sem que haja qualquer possibilidade de interferência direta ou indireta depois de iniciado o processo, estando o praticante da ação "a mercê da sorte".

Assim, com base na redação dada pela legislação, que permite uma ampla interpretação, deduz-se que, na visão do legislador, para se caracterizar um determinar jogo como sendo de azar, deverá haver aleatoriedade suficientemente capaz de impedir a previsão de resultado, mesmo que haja circunstâncias matemáticas que contribuam para o aumento ou redução da probabilidade de determinado resultado ocorrer.

A teoria da probabilidade permite que se calcule a chance de ocorrência de um resultado em um determinado experimento aleatório (Só Matemática, 2011). Assim, se a possibilidade de ocorrência de cada um dos valores do conjunto resultado (casos favoráveis) é a mesma, então a probabilidade de ocorrência de um determinado evento E dentro do espaço amostral, é definida por:

$$P(E) = \frac{n(Casos\ Favoráveis)}{n(Elementos\ do\ Espaço\ Amostral)} \quad (3.1)$$

Experimento aleatório é qualquer experimento cujo resultado depende exclusivamente do acaso, não podendo ser previsto quaisquer de seus valores resultantes, mesmo depois de o

evento ter sido provocado inúmeras vezes sob as mesmas condições (Paes, 2008, p.533).

Com isso, a probabilidade de um determinado valor ser exibido na face superior de um dado não viciado que foi arremessado, é calculada em 1/6, enquanto que a probabilidade de esse valor exibido em sua face superior ser par é de 3/6, pois o objeto possui três valores pares, aumentando a chance de ganho de 16,67% para 50%. Por sua vez, ao desejar dois valores iguais quaisquer ao arremessar dois dados não viciados, a chance de ganho do apostador volta para 16,67%, pois há apenas seis resultados desejados para um conjunto de trinta e seis resultados possíveis.

Apesar de os jogos de azar variar em relação à probabilidade de ganho ou perda, seus resultados, ainda assim, dependem exclusivamente do fator sorte. Para o exemplo descrito no parágrafo anterior, embora cada valor de um dado não viciado apresente probabilidade de 1/6, não há garantia que o valor escolhido seja exibido após seis arremessos.

3.1. MÁQUINAS CAÇA-NÍQUEL (*SLOT MACHINES*)

Originalmente, as máquinas caça-níquel foram desenvolvidas com um conjunto de engrenagens (*reel plate*) e alavancas (*lever*) que acionavam um conjunto de três bobinas (*reel*), cada uma apresentando figuras impressas (*symbol*) em ordem distinta das demais.

Figura 3.1. Componentes de uma máquina caça-níquel mecânica (Merian Webster, 2012).

Ao inserir créditos na máquina (*coin slot*), um sensor liberava a alavanca principal que,

uma vez acionada, movimentava as bobinas em velocidades diferentes, deixando-as girar livremente até que exibissem na face anterior as figuras sorteadas em cada uma delas (Figura 3.1). O pagamento do prêmio está condicionado ao alinhamento de três figuras idênticas na face anterior das bobinas e o seu valor está diretamente ligado à imagem sorteada, conforme previamente definido (*winning line*) na concepção do jogo (Harris, 2011).

Ainda segundo Harris (2011), os projetos originais deram lugar a máquinas controladas ou simuladas por computador, que têm por objetivo causar a mesma impressão dos exemplares mecânicos, embora funcionem com um princípio completamente diferente. Nesses equipamentos, o resultado exibido para cada interação do usuário é controlado por um gerador de números aleatórios (ou pseudoaleatórios), implementado pelo sistema computacional principal.

Nas máquinas controladas por computador, o conjunto de bobinas continua existindo e apenas os processos de acionamento e parada delas sofrem intervenção do sistema computacional.

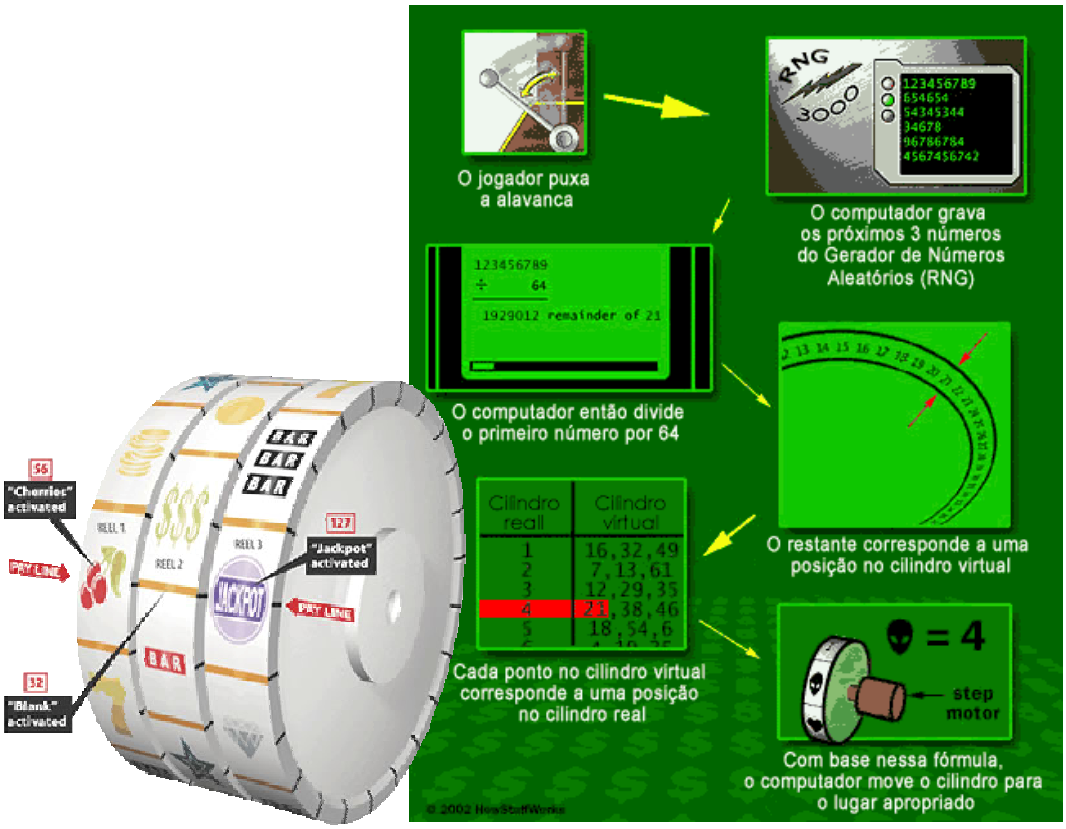

Figura 3.2. Sistema de acionamento das máquinas caça-níquel controladas por computador (Allslotgames, 2012 e Harris, 2011).

Por sua vez, nas máquinas simuladas por computador, as bobinas são meras representações gráficas, submetidas a um processo de animação que remete à forma de funcionamento daqueles dispositivos mecânicos.

Em uma máquina com bobinas contendo 6 figuras diferentes em cada uma e apenas uma sequência de pagamento de premiação, tem-se 216 combinações possíveis e apenas uma combinação desejada, representando uma probabilidade de ganho de 1/216 ou pouco mais de 6%. Assim, para parecerem mais atrativas, as máquinas costumam possuir mais de uma modalidade de pagamento de prêmios menores através de outras combinações de maior probabilidade de ocorrência.

Por não possuírem componentes mecânicos que possam ter sua eficiência constatada em exames periciais, o comportamento operacional de tais máquinas é oferecido por um programa aplicativo carregado no momento de inicialização do equipamento, que recebe interações através de um teclado especial conectado à porta compatível da placa-mãe. Os créditos são inseridos através da porta serial, mediante comandos enviados através de um dispositivo de leitura de fichas, moedas ou cédulas.

A partir do momento em que elementos mecânicos da máquina são substituídos por um programa de computador, deixa de existir transparência quando à forma com que o sorteio é realizado internamente.

Ainda, ao observar de forma superficial essa categoria de máquinas tem-se a impressão de que tais equipamentos são semelhantes aos jogos de fliperama existentes há décadas, onde o jogador interage através de diversos botões e pode influenciar no resultado final a cada ficha inserida, já que tende a adquirir experiência com a forma com que o jogo comporta-se.

Tal semelhança é facilmente identificada através da carcaça externa, geralmente composta por uma caixa de madeira com pintura na cor preta, apresentando em seu terço superior um monitor de vídeo em cores, que pode ser um tubo de raios catódicos (CRT) ou uma tela de cristal líquido (LCD) e diversos botões de grande dimensão, dispostos sobre uma plataforma horizontal ressaltada do gabinete e localizada no terço médio. O dispositivo de leitura de cédulas, moedas ou fichas é comumente localizado no terço inferior do gabinete (Figura 3.3).

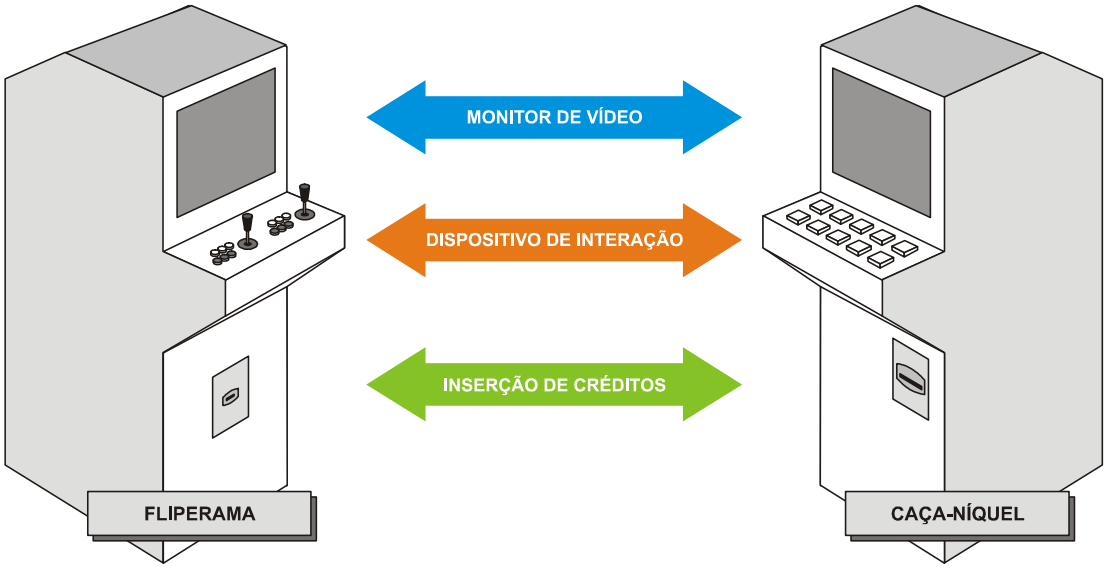

Figura 3.3. Semelhança externa entre as máquinas caça-níquel e as de fliperama.

Não bastassem as reduzidas chances matemáticas de ganho comumente apresentadas por esse tipo de jogo, visto que comumente os programas computacionais de máquinas caça-níquel são projetados e testados para atingir determinada margem de lucro através da retenção de um percentual das apostas (Harris, 2011), os exames realizados pela Seção de Criminalística de Ji-Paraná têm demonstrado que tais programas aplicativos possuem também uma base de informações. Embora haja a necessidade de armazenamento de informações estatísticas referentes à inserção de créditos e pagamentos de prêmios, a relação entre a base de informações e a aplicação não é transparente ao apostador, que geralmente desconhece a existência dela.

Diante dessa ausência de transparência em relação ao mecanismo de sorteio e ao comportamento da máquina diante da base de informações, o emprego de técnicas de Engenharia Reversa de *software* apresenta-se como opção a ser considerada para caracterizar, dentre outras funcionalidades, se o ganho e a perda em tais jogos independem da habilidade do apostador ou se o conjunto de botões da máquina oferece influência no resultado.

Com o suporte de tais técnicas, é possível definir também se os programas computacionais foram projetados (ou sofreram alguma modificação) nos sentido de proporcionar a obtenção de vantagem indevida mediante fraude. Fraude essa que se dá pela minimização das chances originais de ganho do apostador, através da configuração de parâmetros de programação que garantam a obtenção de um lucro pré-definido antes que se pague

qualquer prêmio aos jogadores.

3.2. LEGALIDADE DOS JOGOS DE AZAR

Promulgado em 03 de outubro de 1941, pelo então Presidente da República Getúlio Vargas, o Decreto Lei nº 3688/41 institui a Lei de Contravenções Penais e define em seu art. 50 §3º o jogo de azar como sendo o jogo em que o ganho e a perda dependem exclusiva ou principalmente da sorte do apostador, acrescentando ainda ao rol de tal prática delituosa, as apostas em qualquer modalidade esportiva.

Apesar da lacuna deixada pelo legislador ao não explicitar literalmente as categorias de jogo de azar, percebe-se que a redação da tipificação daquela contravenção penal encaixa-se com o comportamento operacional das máquinas caça-níqueis, pois, desde que não viciadas, apresentam resultado aleatório imprevisível, cuja ocorrência depende exclusivamente do fator sorte, não havendo possibilidade de o apostador influir nessa aleatoriedade do resultado final.

A distinção entre máquinas "viciadas" ou não é importante para a caracterização do poder ofensivo do equipamento. Sendo comprovado que a máquina dispõe de recurso que comprometa a aleatoriedade do resultado, visando reduzir a chance de ganho do apostador, além daquela apresentada pela teoria da probabilidade, não mais será enquadrada como mera contravenção penal (de punição mais branda), mas sim como uma das modalidades do crime de fraude.

No caput do art. 50 fica evidente ainda a intenção do legislador em coibir a exploração de jogos de azar em lugar público ou acessível ao público, prevendo pena de prisão de três meses a um ano e multa, cujo efeito da condenação estende-se inclusive à perda dos mobiliários e demais objetos do estabelecimento infrator (Lei das Contravenções Penais, 1941).

Além de ser uma atividade por si só de natureza ilegal, visto que todas as modalidades de jogos permitidos e executados em território nacional são gerenciadas pela Caixa Econômica Federal, essa modalidade de jogo de azar, além de lesar os cofres públicos, tem como público-alvo a população de baixa renda, já que tais equipamentos vêm sendo instalados em estabelecimentos comerciais destinados a esse público (Nogueira, 2002).

Porém, devido ao fato de os equipamentos atuais possuírem não mais uma alavanca de acionamento, mas um conjunto de botões que passam ao apostador uma falsa impressão de

jogabilidade, tem sido constatada uma divergência de entendimento entre as sentenças prolatadas pelo Poder Judiciário em todo o território nacional (Silva Júnior, 2007) quanto ao enquadramento das máquinas caça-níquel atuais, na legislação especial sobre jogos de azar,

Essa divergência de entendimento, inclusive, permitiu a concessão de liminares para exploração da prática com base em legislações estaduais. Diante disso, a solução plausível encontrada pelo STF para impedir tal discrepância jurídica, foi publicar uma súmula vinculante determinando que somente a União pode legislar a esse respeito por se tratar de prática de competência Federal (Conjur, 2007).

3.3. DEMANDA EM TERRITÓRIO BRASILEIRO

Reforçando o exposto por Nogueira (2002) no que se refere à propagação da exploração de máquinas caça-níquel por todo o território nacional, é possível encontrar inúmeros exemplos de apreensões noticiadas pela mídia impressa e televisionada ao longo dos últimos anos que demonstram a real necessidade de um estudo mais aprofundado do comportamento dos programas de computador responsáveis por simular o ambiente de jogo de azar.

Em novembro de 2007, uma operação conjunta entre a Receita Federal e a Polícia Federal resultou na apreensão de 500 máquinas de videobingo na cidade de Joinville/SC, no depósito de um dos maiores fabricantes desse tipo de máquina no País (G1, 2007).

No interior de São Paulo, em agosto de 2009, depois de percorrer dez cidades a fim de desvendar um esquema de montagem, venda e operações de caça-níquel, a polícia local apreendeu mais de 700 computadores programados para funcionar como videobingo (G1, 2009), enquanto em dezembro de 2011 um salão de beleza de fachada era fechado pela Polícia Civil na cidade de Rio Preto/SP, por abrigar 35 máquinas de videobingo e manter um bingo clandestino (G1, 2011a).

A Polícia Civil do Paraná, em janeiro de 2012 apreendeu 40 máquinas de videobingo em um cassino clandestino que funcionava em uma mansão localizada no bairro Parolin, na capital do Estado (G1, 2012), demonstrando a diversidade de locais em que essas máquinas são encontradas, abrangendo desde pequenos estabelecimentos comerciais como um salão de beleza até grandes imóveis residenciais.

Embora os meios de comunicação de abrangência nacional noticiem com maior ênfase as

apreensões ocorridas nas regiões Sul e Sudeste do País, é possível constatar que tais equipamentos não estão restritos àquelas regiões, já que em dezembro de 2011 as Polícias Civil e Militar do Estado do Pará apreenderam 108 máquinas de videobingo na operação intitulada "Tio Patinhas" (G1, 2011b), enquanto que no Estado de Rondônia, no ano de 2011, foram apreendidas pela Polícia Civil, através da Delegacia de Jogos e Diversões, cerca de 280 máquinas de vídeo bingo em funcionamento (Portal Amazônia, 2011), comprovando a propagação de tal prática delituosa por todas as regiões do País.

Apesar de as máquinas caça-níquel serem probabilisticamente consideradas como jogo de azar, através deste capítulo foi possível verificar que existe uma incerteza acerca do funcionamento interno dos equipamentos atuais, devido ao seu gerenciamento ser realizado por programa de computador de comportamento desconhecido. Esta incerteza permite o surgimento de dúvida quanto à manipulação do fator sorte do apostador e necessita de uma análise aprofundada do código binário de tais arquivos, conforme sugerido no Capítulo 2.

4. ENGENHARIA REVERSA DE SOFTWARE

De acordo com os conceitos vistos no Capítulo 3, verificou-se que as máquinas caça-níquel atuais necessitam de uma análise aprofundada do código binário dos programas que as gerenciam para se ter acesso ao seu modelo comportamental. Tal modelo deve garantir que o seu funcionamento dar-se-á dessa ou daquela forma diante de determinado valor de entrada.

Diante disso, este capítulo apresenta uma síntese das abordagens de Engenharia Reversa, pontuando aspectos técnicos e científicos relevantes ao presente trabalho. Para tanto, além de tratar de peculiaridades da Engenharia Reversa como mecanismos de proteção que visem a dificultar e ferramentas que facilitem a tarefa do engenheiro reverso, aborda também sobre as características comuns existentes entre as arquiteturas tidas como obsoletas e as novas arquiteturas.

Conforme expõe Peters & Pedrycz (2001), "a engenharia reversa é executada com o objetivo de obter uma melhor compreensão de um sistema existente" e "(…) é composta de uma série de técnicas utilizadas para a descoberta de informações a respeito de um sistema de *software*".

A Engenharia Reversa de Software é apresentada por Pressman (2006) como parte integrante de um modelo mais abrangente de Reengenharia de Software. Este modelo é composto por seis etapas, classificadas em dois grupos quanto ao seu objetivo.

No primeiro grupo, responsáveis pela obtenção de um modelo abstrato do comportamento do sistema, estão incluídas as etapas: Análise do Inventário, Reestruturação de Documentos e Engenharia Reversa. Por sua vez, quando for necessária a produção de um produto de *software* aprimorado, desenvolvido com base nos detalhes obtidos durante a análise e que esteja em conformidade com os novos requisitos que se deseja implementar, são empregadas as etapas pertencentes ao segundo grupo: Reestruturação do Código, Reestruturação dos Dados e Engenharia Avante.

Por sua vez, segundo Peters & Pedrycz (2001), o Instituto Americano de Padrões e Tecnologia (NIST) identificou cinco etapas básicas para o processo de Engenharia Reversa, estimando o esforço necessário a cada uma dessas etapas.

Na etapa 1 é definida uma concepção formal da aplicação, consumindo um esforço de pouco mais de 19%.

A etapa 2, por sua vez, abrange todo o processo de extração e análise das informações do aplicativo, garantindo que os elementos do sistema sejam identificados e o código-fonte ou uma representação lógica dele seja obtida. Esta etapa encarrega-se de cerca de 43% do esforço total desprendido na Engenharia Reversa. Alguns dados associados ao sistema, como dados de domínio (entradas e saídas) e dados de controle (decisão e iteração), também são identificados, permitindo uma visão abstrata do modelo de *software* analisado e sobre as suas regras gerais de negócio.

Por fim, as etapas 3, 4 e 5 dizem respeito à Engenharia Progressiva, que versa sobre a criação de um novo sistema com base na combinação de novos e antigos requisitos e que não serão aqui pormenorizados, visto não tratar do escopo principal deste trabalho, que tem como foco principal a etapa 2 do modelo proposto pelo NIST.

Assim, é prudente salientar que o escopo do trabalho proposto abrange somente as etapas 1 e 2 do modelo proposto pelo NIST, que representam cerca de 62% do esforço necessário de um processo integral de Reengenharia de Software.

4.1. ÁREAS DE APLICAÇÃO

Tendo aplicações diversas, a Engenharia Reversa de Software, segundo Vênere (2009, p.8) vem auxiliando na análise de programas maliciosos, principalmente pelas empresas desenvolvedoras de aplicativos de proteção contra tais “pragas”; no desenvolvimento de programas para controle de dispositivos (*driver*) para ambientes não apreciados pela indústria que o desenvolveu; no entendimento do funcionamento de um programa proprietário, para posterior desenvolvimento de uma versão livre que não se utilize do mesmo código daquele analisado; no entendimento do funcionamento de produtos existentes para determinação de protocolos de compatibilidade que deverão ser seguidos durante o desenvolvimento de produtos futuros; e documentação de programa legado do qual não se tenha mais o código-fonte.

Corroborando com o entendimento de Vênere (2009), Pressman (2006), defende que a Engenharia Reversa de Software pode ser executada sobre os dados, sobre as funcionalidades ou sobre a sua interface, através da análise das entradas e saídas geradas durante a utilização do sistema. É possível ainda ser executada sobre esses três itens em conjunto, correlacionando-os de forma a possibilitar a geração de um modelo abstrato do produto analisado, que apresente um alto grau de fidelidade ao original.

Ao relacionar o exposto por ambos os autores é possível supor que, se aplicada sobre o programa de computador utilizado por equipamentos eletrônicos que simulam um ambiente de aposta de jogo de azar, a Engenharia Reversa pode resultar em um modelo comportamental do sistema que permita estabelecer um paralelo com as práticas delituosas previstas na matéria Penal.

Desta forma, ao submeterem-se os dados ao processo de Engenharia Reversa, objetiva-se o entendimento tanto das estruturas de dados internas, focalizando a definição de uma classe de objetos com seus tipos abstratos de dados, como da estrutura do banco de dados, independentemente da sua organização lógica e estrutura física.

Sendo aplicada sobre as funcionalidades do sistema, a Engenharia Reversa tem como objetivo primordial entender o seu processamento, abstraindo aqueles procedimentos que foram codificados durante o desenvolvimento e que posteriormente, na fase de compilação do código-fonte, foram convertidos em linguagem de máquina. No caso em lide, é possível utilizar tal característica para verificar a utilização de elementos que evidenciem as práticas que se deseja comprovar, tais como funções de geração de números aleatórios na ação de sorteio ou comportamentos condicionados a valores armazenados na base de dados.

Na aplicação da Engenharia Reversa sobre a interface, esta permite que se estabeleça um modelo comportamental do sistema, diante da resposta apresentada para cada uma das ações básicas de interação. Tal aspecto da Engenharia Reversa mostra-se de extrema importância na análise dos equipamentos caça-níquel em questão, pois a partir dele é possível relacionar o comportamento real do equipamento, constatado quando da análise das funcionalidades, com o comportamento que o apostador é induzido pela interface a esperar.

Para Vênere (2009, p.8) o processo de Engenharia Reversa pode envolver diversas técnicas, abordando desde a execução do aplicativo em um ambiente controlado, para compreensão das ações e modificações diante das interações sofridas, como também a análise do código de máquina, com o objetivo de traduzir suas funções em um nível mais alto de código.

4.2. ABORDAGENS

Dentre as abordagens existentes para a Engenharia Reversa, a escolha de uma delas está diretamente ligada às características do programa que se deseja analisar.

Uma vez compilado um código-fonte de um determinado programa, seu conteúdo é traduzido em uma linguagem de baixo nível, que será compreendida pelo Sistema Operacional desde que pertencente à mesma plataforma de desenvolvimento. Assim, a abstração do código original é substituída por instruções da plataforma escolhida que serão executadas pelo *hardware*.

Diante disso, a escolha da abordagem deve tomar como base a plataforma na qual o programa foi desenvolvido, a técnica ou ferramenta utilizada no seu desenvolvimento e o tipo de informação que se deseja extrair.

Para Eilam (2005, p.110), há duas abordagens de análise, sendo elas a "análise *off-line*" e a "análise *online*". De acordo com as características descritas no parágrafo anterior e com base em outras que serão observadas quando de uma análise prévia do código *assembly* gerado a partir do arquivo executável, as abordagens poderão ser utilizadas separadamente ou em conjunto.

Na abordagem denominada "análise *off-line*" de código, tida por Eilam (2005, p.110) como a mais avançada abordagem para Engenharia Reversa, partes do código *assembly* são lidas e analisadas manualmente com o intuito de compreender funcionalidades isoladas, antes de correlacioná-las, necessitando de uma melhor compreensão de todo o código, uma vez que é muito difícil determinar completamente o fluxo de execução.

4.3. PROTEÇÕES CONTRA ENGENHARIA REVERSA

Embora haja casos em que um programa computacional empregue, de forma intencional, técnicas que visem à proteção contra Engenharia Reversa (Eilam, 2005, p.327), em outras circunstâncias, algumas técnicas são empregadas única e exclusivamente com o objetivo de melhorar alguma característica do código binário.

Em equipamentos com capacidade reduzida de armazenamento permanente e de memória de execução, é prática comum a adoção de técnicas de empacotamento de arquivos binários. Essas técnicas possuem a propriedade de reduzir o tamanho final do arquivo, chegando a uma taxa de compressão de até 70%, sem que seja necessária qualquer modificação no código ou adaptação na hora da execução (Hardware.com.br, 2007).

De acordo com as especificações de cada plataforma suportada, o empacotador adiciona um pequeno módulo no início do arquivo. Em casos onde o código é empacotado, este somente tornar-se-á legível quando da sua execução, pois o módulo inserido no início do

código binário realizará a tarefa de extração do conteúdo do arquivo para a memória e em seguida passará o controle a ele.

Para esses casos, por exemplo, é necessária a utilização da abordagem de análise *online*, onde, com o auxílio de um conjunto de ferramentas apropriadas a este tipo de análise, permite que o analista verifique a implicação de uma instrução em relação aos dados (e vice-versa) manipulados pelo programa analisado.

Além do empacotamento do código, Vênere (2009, p.106) aponta ainda outras técnicas utilizadas por desenvolvedores para evitar a realização da Engenharia Reversa de seus produtos de *software*, tais como "detecção de *debuggers*, modificação de exceções, detecção de ambiente virtual, entre outras técnicas", que deverão ser observadas quando da análise *online*, pois podem comprometer o resultado final produzido por esta abordagem.

4.4. FERRAMENTAS

Conforme preceitua Eilam (2005, p.14), durante a etapa de Engenharia Reversa são necessárias ferramentas das mais diversas naturezas, sendo impossível a sua realização sem elas. Este conjunto de ferramentas necessárias abrange desde aquelas que não foram desenvolvidas para esta finalidade, tais como as de monitoramento do sistema operacional, como aquelas produzidas exclusivamente para: traduzir as instruções de máquina para linguagem *assembly* (*disassemblers*); executar passo a passo o código (*debuggers*); traduzir a linguagem de máquina para alguma linguagem de desenvolvimento de alto nível (*decompilers*) (Eilam, 2005, p.109).

O *disassembler* é uma das principais ferramentas do processo de Engenharia Reversa, pois decodifica a linguagem de máquina, composta por instruções computacionais representadas por códigos binários ininteligíveis, em instruções da linguagem *assembly* pertencentes à arquitetura e à plataforma para as quais o programa foi compilado. Apesar da sua complexidade, *assembly* é uma linguagem textual, sendo mais facilmente compreendida que as instruções binárias geradas pelo processo de compilação do código-fonte original.

Dentre as ferramentas amplamente utilizadas, destaca-se a IDA Pro[3] na categoria

[3] www.hex-rays.com/products/ida/support/download_freeware.shtml

disassemblers e as ferramentas OllyDBG[4] e SoftICE[5] na categoria *debuggers*. Estas últimas, apesar de oferecem também a funcionalidade de *disassembler*, apresentam menos recursos.

Devido a um conjunto de funcionalidade apresentadas pela ferramenta IDA Pro, que tornam a tarefa de Engenharia Reversa mais amigável, a versão 4.9, de distribuição gratuita, foi previamente sugerida como candidata no processo de análise *off-line* de código. Dentre estas funcionalidades, destaca-se o suporte a uma grande variedade de formatos de arquivos executáveis e de arquiteturas de processadores, sendo esta última de fundamental importância para tradução correta das instruções em *assembly*, já que cada arquitetura possui instruções particulares (Eagle, 2008, p.5)

A ferramenta OllyDBG, tida como o melhor *debugger* em nível de aplicação (Pirigov, 2006, p.174), apresentou-se como candidata prévia à tarefa de análise de código *online*, devido ao fato de se tratar de uma ferramenta gratuita e de a ferramenta SoftICE ter sido descontinuada pelo seu fabricante, não sendo optado pela realização de qualquer espécie de prova de conceito devido a predominância de ambas as ferramentas dentre as referências bibliográficas consultadas.

Porém, embora ambas as ferramentas tenham se apresentado como candidatas naturais para desempenhar as tarefas a que se propõem, a IDA Pro teve seu uso limitado no estudo, enquanto que a OllyDBG foi completamente descartada devido ao fato de ser voltada para arquiteturas de 32-bits, não suportando a arquitetura de 16-bits na qual o programa das máquinas caça-níquel foi compilado.

Diante dessa limitação, optou-se pela utilização do *debugger* CodeView[6], que apesar de não oferecer recursos avançados de análise *online*, foi concebido para a arquitetura de 16-bits.

4.5. ARQUITETURAS

Embora tenha havido outros conjuntos processadores antecessores ao da família 8086, este foi o que alavancou a evolução dos equipamentos de informática ao longo dos anos, tal qual são conhecidos hoje. Os processadores da família 8086 oferecem uma arquitetura

[4] www.ollydbg.de/
[5] Ferramenta descontinuada pela empresa NuMega
[6] *Debugger* independente desenvolvido por David Norris

interna com capacidade de processamento de instruções de até 16 bits, substituindo (porém mantendo compatibilidade) a arquitetura anterior de 8 bits da família 8085, que já não suportava a necessidade de ampliação de armazenamento interno de instruções. Segundo Zelenovsky & Mendonça (2006, p.24) a família de processadores 8086 guarda "tanto em *hardware* quanto em *software*, uma grande semelhança com o 8085", onde "o conjunto dos registradores do 8086 é um superconjunto dos registradores do 8085".

Essa mesma abordagem utilizada para criar uma arquitetura de 16 bits que superasse a anterior, mas mantivesse a compatibilidade, foi utilizada quando do surgimento da família 386 e das demais que se seguiram, com arquitetura interna de 32 bits (Clube do Hardware, 2001). Tal abordagem é utilizada também atualmente pelas arquiteturas de 64 bits que suportam os conjuntos de instruções e os registradores das arquiteturas anteriores.

Para Veloso (2003, p.25), os registradores são dispositivos internos de armazenamento que possibilitam a execução das tarefas a que o conjunto processador propõe-se, já que segundo Vênere (2009, p.67) este não tem acesso direto à memória principal e necessita primeiramente copiar o conteúdo desejado para a sua memória interna, representada pelos seus registradores, antes de realizar o devido processamento.

Embora a quantia de registradores de um processador varie entre "uns poucos e mais de uma centena" de acordo com o seu modelo (Veloso, 2003, p.25), as arquiteturas de 16 e 32 bits, por exemplo, possuem um conjunto de 08 (oito) registradores de propósito geral (Eilam, 2005, p.44). Esse conjunto de registradores é utilizado quando da conversão da linguagem de alto nível, usualmente empregada para escrever programas de computador, em linguagem de máquina (*assembly*) durante o processo de compilação.

Dos registradores de propósito geral da arquitetura de 32-bits, os quatro primeiros (EAX, EBX, ECX e EDX) são definidos por Pirigov (2006, p.22) como registradores de trabalho, pois apresentam uma forma de subendereçamento, onde é possível manipular partes específicas de cada registrador utilizando-se dos nomes dos registradores correspondentes nas arquiteturas de 16 e 8 bits.

Os dezesseis primeiros bits do registrador EAX, o qual tem a função de acumular valores diversos, inclusive aqueles retornados por funções, podem ser acessados através do nome do registrador AX, seu correspondente na arquitetura de 16 bits. Da mesma forma, os oito primeiros bits e os oito bits seguintes podem ser acessados através dos nomes dos registradores AL e AH, respectivamente. Na arquitetura de 64 bits, por sua vez, o

registrador correspondente ao EAX é denominado RAX, conforme ilustrado na Figura 4.1 a seguir.

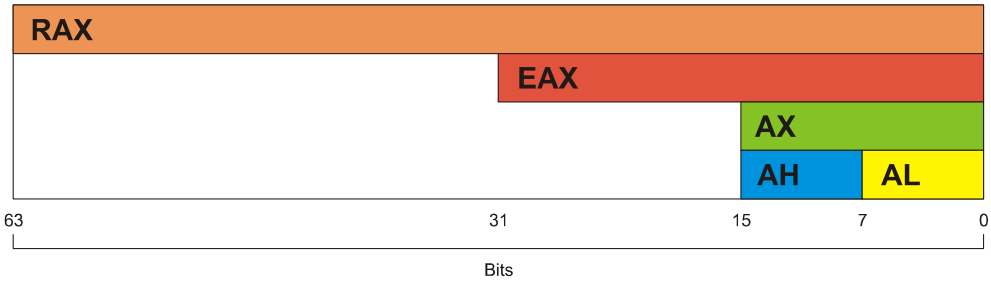

Figura 4.1. Endereçamento de registradores em arquiteturas de 32 bits.

Em ambas as abordagens da Engenharia Reversa por análise de código é de fundamental importância ter esse conceito muito bem compreendido, pois durante a análise é costumeiramente "motivo de confusão", conforme garante Vênere (2009. p.68). Uma vez que se tenha em mente as atribuições e formas de endereçamento de cada registrador, determinados trechos de código de máquina podem ser mais facilmente compreendidos. O registrador ECX da arquitetura de 32 bits, por exemplo, é comumente empregado na contagem de saltos de uma função iterativa, enquanto que o registrador EBX incumbe-se da tarefa de endereçamento indireto de dados e comumente recebe o endereço-base de um determinado objeto. O registrador EDX, por sua vez, encarrega-se do armazenamento das demais informações de uso geral que se esteja manipulando.

Apesar das atribuições de cada registrador, algumas vezes o compilador pode empregá-los de forma distinta daquela a que se destina, caso necessite de mais registradores para converter uma determinada instrução escrita em linguagem de alto nível.

Durante a realização da Engenharia Reversa de uma aplicação através da análise de código *online*, a compreensão da função dos registradores de índice (ESI e EDI) e ponteiro (EBP e ESP) também é de suma importância. Os primeiros são frequentemente empregados como índices de origem (*source index*) e destino (*destination index*) em instruções responsáveis pela cópia byte a byte do conteúdo de um determinado endereço de memória. Os registradores de ponteiro são empregados para marcar na memória a base (*base pointer*) da zona de abrangência de uma determinada variável em relação à pilha de execução, assim como o endereço que aponta para o topo dessa pilha (*stack pointer*) e que impede que os valores de endereços inferiores a ele sejam acessados sem que o ponteiro movimente-se com o desempilhamento de valores (Eilam, 2005, p.45).

Embora o suporte a arquiteturas anteriores seja importante para manter a compatibilidade de um conjunto processador com as aplicações escritas em outras arquiteturas, tal característica apresenta-se como fator complicador quando da realização da Engenharia Reversa de uma aplicação escrita e compilada em arquiteturas tidas como obsoletas. Ferramentas atuais destinadas a este fim como a IDA Pro podem realizar de forma incorreta, em uma abordagem *top-down*, a tradução das instruções com base na quantia de bytes da arquitetura a qual dá suporte.

Assim, ao submeter um aplicativo desenvolvido e compilado em uma arquitetura de 16 bits a uma ferramenta cujo suporte abrange somente a arquitetura de 32-bits, as instruções formadas por dois bytes não serão corretamente identificadas. Ela interpretará o código de forma incorreta tentando converter uma instrução a partir de quatro bytes ao invés dos dois originalmente utilizados. O mesmo fato ocorre com as instruções de uma aplicação desenvolvida em arquitetura de 64-bits e que sejam submetidas à ferramenta em questão.

Essa peculiaridade de análise de código, somada ao fato de os jogos aparentemente terem sido desenvolvidos em plataforma de 16-bits, dificultou o processo de Engenharia Reversa das máquinas caça-níquel, pois as ferramentas disponíveis atualmente foram desenvolvidas com o objetivo de fornecer funcionalidades destinadas à facilitação da tarefa de Engenharia Reversa de código recente (32-bits ou 64-bits).

Restando as ferramentas tidas como obsoletas, que são fracas no suporte à depuração de código, os procedimentos de análise da aplicação principal das máquinas caça-níquel foram dificultados de sobremaneira.

Neste capítulo, mais do que uma discussão acerca das abordagens de Engenharia Reversa ou das ferramentas empregadas para facilitar a análise, foi possível verificar que a compatibilidade existente entre as arquiteturas de 16, 32 e 64-bits, quanto às instruções e à forma de endereçamento dos registradores, sugere a possibilidade de emprego da metodologia de análise demonstrada no Capítulo 6 em equipamentos de arquiteturas distintas daquela dos equipamentos que compõem o objeto de estudo.

5. TRABALHOS CORRELATOS

Apesar de as máquinas caça-níquel simuladas por programas de computador existirem há algumas décadas e continuarem a utilizar componentes obsoletos oriundos de computadores da plataforma PC e técnicas de desenvolvimento de *software* em desuso atualmente, são raros os trabalhos específicos ao tema.

Tal precariedade em relação à quantidade de material existente em língua inglesa supostamente deva-se ao fato de que em vários estados americanos o jogo de azar é permitido, não havendo assim a necessidade ou o interesse em realizar investimentos que visem à descoberta do comportamento interno de tais equipamentos.

Por sua vez, no que se refere à escassez de material existente em língua portuguesa, supõe-se que tal fato deva-se à especificidade do tema, que é de natureza estritamente pericial, em detrimento da própria complexidade apresentada pelas técnicas de Engenharia Reversa e da escassez de mão de obra especializada disponível em território nacional tanto em ambiente acadêmico quanto comercial.

Diante de tal fato, é apresentada como trabalho correlato a análise estatística de máquinas caça-níquel, realizada pelo Instituto Nacional de Criminalística da Polícia Federal (Nogueira, 2002), para demonstrar que tal abordagem é inviável de ser aplicada à categoria de máquinas abrangidas por este estudo.

A aplicação da Engenharia Reversa para compreensão de código malicioso (Vênere, 2009) é apresentada como uma metodologia de análise de arquivos binários, onde são buscadas primeiramente as informações de mais fácil acesso.

Para comparação de arquivos binários referentes a jogos aparentemente distintos, é citada a comparação isomórfica de objetos executáveis baseada em grafos de Dullien & Rolles (2005) e Flake (2004).

5.1. ANÁLISE ESTATÍSTICA DE MÁQUINAS DE VÍDEO BINGO

O trabalho realizado por Nogueira (2002) teve como objetivo a proposta de uma metodologia para realização de exames em máquinas caça-níquel com enfoque puramente estatístico, utilizando-se de estudos de caso realizados sob o prisma da criminalística.

O primeiro caso de uso abordado no trabalho trata de uma máquina de fabricação espanhola, do fabricante Unimesa, dotada de três cilindros mecânicos independentes que

apresentam diversas figuras para premiação mediante combinação por alinhamento, acionados por uma alavanca lateral, cuja modalidade de aposta ou de jogada é escolhida por um teclado composto por cinco botões. A alimentação do equipamento dá-se por dispositivos de leitura de ficha e moeda ou cédulas presentes na face anterior e lateral do equipamento, respectivamente.

Para o segundo caso, o autor utiliza um equipamento misto de procedência nacional, no modelo de roleta eletrônica, do fabricante São Francisco, que não trabalha com sistema de fichas ou cédulas, mas sim com o pagamento de créditos diretamente ao estabelecimento detentor do equipamento, permitindo a participação de até 06 (seis) apostadores simultaneamente, além do controlador da mesa. O equipamento é composto por uma roleta eletrônica, manipulada pelo controlador da mesa através de um conjunto de 12 (doze) botões e por um monitor de vídeo que simula o tabuleiro de uma roleta real. Cada apostador dispõe de um conjunto de 05 (cinco) botões distintos para interagir com o jogo no que se refere à escolha das jogadas ou apostas.

Tabela 5.1. Estatísticas de Jogadas.

Modelo	Ganhos	Perdas	% Ganhos	% Perdas
Tycoon	20	110	15,38	84,62
Twin Sparks	6	87	6,45	93,55
Joker's Wild	18	42	30,00	70,00
Deuce's Wild	18	45	28,67	71,43
Roleta Eletrônia	22	60	26,83	73,17

Para o primeiro caso de uso, envolvendo os jogos "Tycoon", "Twin Sparks", "Joker's Wild" e "Deuce's Wild", observa-se que o autor obteve uma probabilidade de ganho entre 6,5% e 30%. No segundo caso de uso, envolvendo o jogo "Roleta Eletrônica", por sua vez, foi obtido um percentual de ganho próximo a 27, conforme Tabela 5.1. O autor não apresenta em seu estudo qualquer relação do ganho com a tabela de premiação das máquinas, já que as ocorrências de ganho podem envolver desde prêmios mínimos até prêmios máximos. Diante disso, percebe-se que o estudo demonstra tão somente o percentual de resultados favoráveis e não favoráveis dos equipamentos, não refletindo o valor líquido proveniente da relação apostas x prêmios.

Observa-se ainda que a pesquisa realizada por Nogueira (2002) não aborda qualquer tentativa de identificação de fraude no sistema de sorteio das jogadas. Embora o autor garanta que as máquinas "Tycoon" e "Twin Sparks", que possuem mecanismo interno de funcionamento semelhante, sejam controladas por programa de computador, o resultado

estatístico das jogadas demonstra uma diferença de aproximadamente 8,9%. Uma vez que nessa categoria de equipamentos o valor sorteado (aleatoriamente) é responsável por acionar os sensores que fazem cada um dos cilindros posicionarem-se na figura correspondente, a diferença apresentada pelos dois exemplares pode indicar uma possível fraude, supostamente gerada pela existência de parâmetros de comportamento que possam ser configurados com valores distintos.

Conforme visto no Capítulo 3, a probabilidade de ganho de uma máquina caça-níquel é definida pela razão da quantia de casos favoráveis pela quantia de elementos do espaço amostral, não justificando tamanha margem de erro e deixando dúvidas quanto à forma interna de sorteio de tais programas, principalmente no que se refere à existência ou não de um processo iterativo.

5.2. ENGENHARIA REVERSA DE MALWARE

No trabalho apresentado por Vênere (2009, p.118-143), depois de discorrer sobre algumas áreas do conhecimento que permeiam a atividade de Engenharia Reversa, o autor fornece uma metodologia de análise de um código malicioso, sugerindo, devido à complexidade que o tema oferece, "onde um binário desconhecido pode ser tão complexo que o analista poderia levar meses realizando essa tarefa", que a análise inicie-se a partir dos itens considerados mais fáceis, onde podem ser encontradas informações mais prontamente acessíveis, sem que seja necessário um grande desprendimento de tempo de análise *off-line* de código.

Dentre os itens elencados, o primeiro deles é a busca por cadeias de caracteres no arquivo executável, cuja finalidade é encontrar vestígios de comportamento da aplicação que possam ser utilizados como ponto de partida quando de uma análise mais aprofundada.

Para o autor, outra informação de suma importância é a detecção de funções importadas pela aplicação, já que algumas delas referem-se à leitura de teclado, tratamento de cadeias de caracteres, criação e abertura de arquivos, dentre outras, além do fato de que aplicações em ambiente Windows compartilham funções daquele sistema operacional que são amplamente conhecidas pelos desenvolvedores de *softwares* e que integram a sua API (*application programming interface*).

Como o autor utiliza-se da ferramenta IDA, este sugere a utilização da sua funcionalidade de detecção de código-padrão, evitando analisar código que já tenha sido previamente

identificado pela ferramenta como sendo pertencentes à biblioteca do sistema.

A análise do código binário é sugerida tão somente depois de examinados os nomes de estruturas que tenham sido identificadas automaticamente pela ferramenta de análise, bem como os gráficos de chamadas de funções, sendo essa última realizada depois de renomeadas aquelas cujo sentido é possível identificar, já que facilitam a compreensão do relacionamento existente entre elas e consequentemente, do comportamento geral da aplicação.

Uma vez realizada a análise *off-line*, com base nas informações coletadas preliminarmente, proceder-se-á à realização de uma análise ao vivo, onde a aplicação esteja executando em um ambiente controlado que permita monitorar o seu comportamento sem comprometer a segurança do sistema hospedeiro.

Assim, definida a metodologia de análise, o autor demonstra a eficácia da sua aplicação em uma das variantes do *worm* denominado MyDoom, buscando identificar: sua forma de propagação; dados que porventura sejam acessados ou modificados; recursos utilizados para manter-se no sistema; existência de *backdoor* que permitam o acesso por aplicações externas; comportamento da aplicação depois de infectado o sistema hospedeiro e; prejuízo causado ao equipamento infectado em relação a sua presença.

Para responder ao questionamento quanto à forma de propagação do código malicioso, depois de realizada uma busca nas cadeias de caracteres extraídas do código executável, bem como na lista de funções importadas pela aplicação, o autor identifica a existência de informações relacionadas à criação de mensagens eletrônicas, sugerindo como forma de propagação o envio através de mensagens em massa (SPAM). Fato este comprovado pela identificação de funções de criação de arquivo anexado, geração de data e hora, geração de endereço IP, geração de cabeçalho de mensagem, geração de assunto e nome de arquivo anexado, dentre outras.

Em seguida, o autor novamente ampara-se na relação de funções utilizadas pela aplicação, para identificar aquelas responsáveis pela criação e acesso de arquivos e diretórios ou pelo acesso ao registro como CreateFileA(), FindFirstFileA() e RegOpenKeyExA(), por exemplo, onde consegue constatar que o código malicioso realiza uma busca no registro do Windows com o intuito de encontrar o diretório do cliente de mensagem eletrônica Outlook e assim ter acesso à lista de contatos do usuário, bem como constata que a aplicação cria uma chave de registro que possibilita a execução automática do código

malicioso quando da inicialização do sistema, representando a forma que a aplicação utiliza para se manter ativa no sistema hospedeiro mesmo depois do seu desligamento.

Durante a análise, o autor identifica ainda que o código malicioso traz consigo um programa embutido que abre uma vulnerabilidade (*backdoor*) no sistema hospedeiro e permite o acesso remoto por parte do autor da aplicação, depois de comprometido o sistema, mediante a criação de novos processos em tempo de execução, conforme evidenciado pela utilização da função CreateProcessA().

Com isso, sem que houvesse a necessidade de uma análise aprofundada do código binário, o autor, utilizando-se de funcionalidades da ferramenta IDA, comprovou a possibilidade de se extrair informações de grande importância para a compreensão do comportamento interno de uma determinada aplicação maliciosa, garantindo a eficácia do método proposto por ele e sugerindo, por analogia, a possibilidade de êxito também quando do seu emprego em programas de computador sob os quais pairem dúvidas acerca do seu comportamento interno.

5.3. ANÁLISE DO ISOMORFISMO DE OBJETOS EXECUTÁVEIS

Diante da dificuldade de identificar os pontos de discrepância entre dois objetos executáveis semelhantes através da análise direta do código, Dullien & Rolles (2005) e Flake (2004) apresentam uma abordagem de comparação de dois programas executáveis quaisquer através do mapeamento *one-to-one* das funções identificadas no código.

A comparação não aprofunda ao nível de detalhamento das instruções *assembly*, restringindo-se tão somente ao grafo de controle de fluxo (cfg) das funções. Com isso, divergências de código referentes ao reordenamento de instruções e a mudanças na alocação de registros, não produzirão resultados falso-negativos.

Esta abordagem implica que funções, cuja funcionalidade tenha sido alterada de forma significante, não sejam mapeadas, possibilitando a fácil identificação de mudanças comportamentais existentes entre duas versões de um mesmo programa.

Ao contrário das abordagens de comparação baseadas na sequência de linhas do código-fonte, os autores adotam uma análise centrada em grafos, desprezando tanto código *assembly* quanto for possível. Nesta abordagem, o programa executável é representado por um conjunto de funções $F = \{f_1,\ldots, f_n\}$. O relacionamento entre essas funções é representado por um grafo direcionado, onde cada elemento de F é um nodo e as chamadas

de função são arestas que conectam f_i a f_k.

Toda função $f_i \in F$ pode ser representada como um grafo de controle de fluxo (*control flow graph - cfg*), que consiste de blocos básicos de controle e o relacionamento existente entre eles. Os blocos básicos de controle correspondem às funcionalidades básicas de um determinado trecho de código da função.

Assim, dados dois programas executáveis A e B, é criado um mapeamento bijetivo p entre os elementos dos conjuntos de funções $A = \{a_1,\ldots, a_n\}$ e $B = \{b_1,\ldots, b_m\}$.

$$p : \{a_1,\ldots, a_n\} \rightarrow \{b_1,\ldots, b_m\} \tag{5.1}$$

Para o mapeamento, são consideradas três informações de um nodo: a) o número de blocos básicos de controle; b) o número total de arestas; c) número de arestas que são originadas a partir daquele nodo. Porém, o mapeamento ocorre somente se não existirem mais nodos em A ou B que convirjam com essas informações (Figura 5.1).

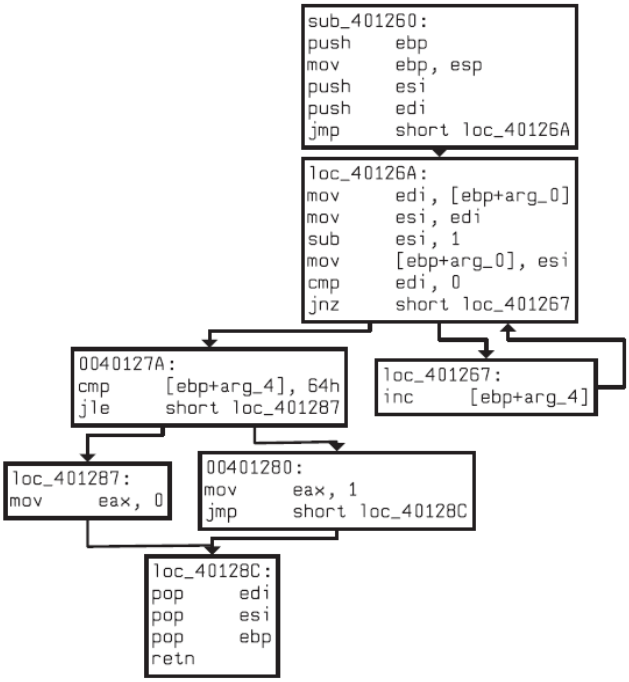

Figura 5.1. Exemplo de grafo de controle de fluxo da função “sub_401260”.

O método além de ser livre de falso-positivos, apresenta uma boa independência das instruções *assembly* de uma determinada família de processadores, já que a única dependência que deve ser considerada é a capacidade de distinguir entre uma chamada de subfunção e outras chamadas quaisquer.

Através dos trabalhos correlatos apresentados foi possível verificar que análise estatística

realizada por Nogueira (2002) é inviável de ser aplicada nas máquinas caça-níquel atuais, devido à grande quantidade de combinações de interação possibilitada pelos equipamentos. Ao invés disso, fica evidente que a melhor abordagem para a elucidação do comportamento dos programas de computador que gerenciam essas máquinas é o emprego da Engenharia Reversa, adaptando a metodologia apresentada por Vênere (2009) para análise de código malicioso ao modelo proposto pelo NIST e descrito no Capítulo 4.

Por sua vez, a comparação isomórfica de executáveis de Dullien & Rolles (2005) e Flake (2004) apresentou-se de grande relevância para comprovar que aplicações de máquinas aparentemente distintas possuem, em sua essência, o mesmo comportamento geral.

6. ANÁLISE DE MÁQUINAS CAÇA-NÍQUEL

Embora tenha sido demonstrado no Capítulo 3 que as apreensões de máquinas caça-níquel vêm ocorrendo em todo o território brasileiro ao longo dos anos, o presente estudo foi realizado com base nos equipamentos arrecadados dentro da circunscrição da Delegacia Regional de Polícia Civil de Ji-Paraná, que abrande 09 (nove) dos 54 (cinquenta e quatro) municípios do Estado de Rondônia.

Vislumbrando a necessidade de realizar um estudo mais aprofundado sobre o comportamento interno de tais aplicações, os Peritos Criminais da área de Computação Científica (área do conhecimento inserida no quadro funcional da Polícia Civil do Estado de Rondônia no ano de 2005), a partir no ano de 2009, sempre que solicitados a examinar equipamentos dessa natureza, passaram a recolher os dispositivos de armazenamento permanente de cada máquina com o intuito de compor um acervo das aplicações comumente utilizadas por elas, informando tal fato no fechamento do respectivo laudo para conhecimento dos órgãos de Justiça.

Assim, este capítulo descreve a adaptação do método proposto por Vênere (2009) ao modelo sugerido pelo NIST, realizando a análise primeiramente da usabilidade da interface das máquinas caça-níquel, por retornar mais facilmente um modelo inicial de comportamento, seguida da análise dos itens de *hardware*, com o objetivo de identificar os elementos de entrada e saída de tais equipamentos. Uma vez obtido o modelo inicial de comportamento, é realizada a análise do ambiente operacional, que é responsável por preparar os itens de *hardware* e carregar a aplicação principal.

Na sequência, através da análise interna da aplicação principal, é verificada a consistência do modelo inicial de comportamento, obtido na análise da usabilidade da interface, em relação ao tratamento das informações de entrada e as informações de saída apresentadas ao apostador. Nesta etapa são analisadas possíveis tentativas de ofuscamento e dissimulação de informações.

Dentre as máquinas comumente apreendidas, são encontradas tanto as compatíveis com o estudo ora realizado, como também são encontradas máquinas que possuem *hardware* específico e não dispõem de monitor de vídeo, mas sim uma placa de circuito impresso dotada de diversos LEDs que indicam resultados em um painel estático ilustrado. Como dito na definição do escopo do trabalho, tais equipamentos não foram incluídos no estudo

de caso realizado.

Essas últimas, devido ao aspecto dos circuitos integrados presentes na placa do painel principal e às características dos demais itens de *hardware*, remetem a uma data de fabricação anterior à dos demais equipamentos.

Entretanto, em meio às máquinas que apresentam características semelhantes às daquelas que representam o objeto de estudo do presente trabalho, são encontrados exemplares que, ao invés de utilizarem dispositivos de armazenamento conectados ao barramento IDE da placa-mãe, empregam *hardware* específico. Este *hardware* específico é composto por placas de circuito impresso que possuem o jogo gravado em um circuito integrado afixado a um dos seus *slots* e são conectadas aos barramentos ISA ou PCI.

Analisando o grau de complexidade em se criar tais placas de circuito impresso, correlacionando tal resultado com o fato de os barramentos ISA e PCI estarem cada vez mais obsoletos, somado ainda à redução no porte de tais equipamentos (Figura 6.1) e ao barateamento dos dispositivos de armazenamento atuais que se utilizam de barramentos como IDE, SATA e USB, deduz-se que uma suposta evolução das máquinas caça-níquel encaminha-se cada vez menos para *hardwares* específicos. Consequentemente, a análise comportamental das aplicações principais das máquinas abrangidas pelo presente estudo tende a ser aplicável a um conjunto significativo de máquinas atuais e em razoável tempo futuro.

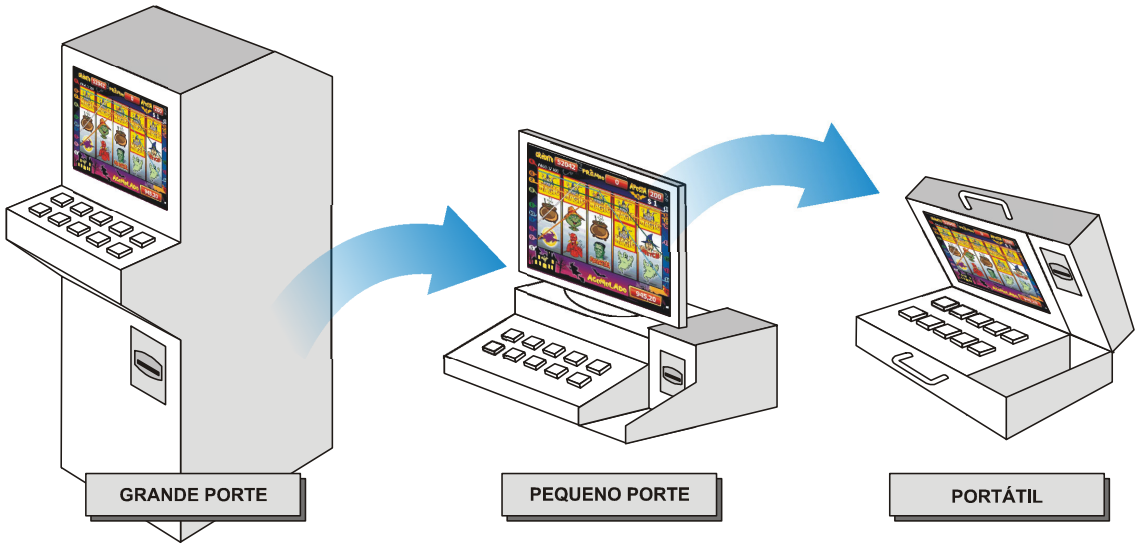

Figura 6.1. Evolução das máquinas caça-níquel quanto ao porte.

Assim, excluindo-se os dispositivos de armazenamento que apresentaram falha na leitura

do seu conteúdo, motivados por fatores diversos, como transporte e acondicionamento incorretos, o presente estudo foi realizado com 08 (oito) exemplares de máquinas caça-níquel, as quais abrigam aparentemente 18 (dezoito) versões distintas da aplicação principal.

Tabela 6.1. Relação de jogos oferecidos pelos equipamentos analisados.

	Capacidade de Armazenamento	Prêmios Acumulados	Jogos oferecidos
Máquina 1	256 MB	1	Golden Cards
		1	Halloween I
		1	Halloween Mais Forte
		1	Pantanal
		1	Trago Tour
		1	Vaka Loka
Máquina 2	512 MB	2	Halloween II
		2	Sexy ++
Máquina 3	512 MB	1	Golden Cards
		2	Campeonato Brasileiro II
		2	Halloween II
		2	Halloween Trick
		1	Halloween Mais Forte
Máquina 4	512 MB	2	171 Forte
		2	Bucaneiros
		2	Halloween II
		2	Oro de Mexico 2
		2	Vaka Loka
Máquina 5	128 MB	1	Halloween Mais Forte
Máquina 6	256 MB	1	171 Forte
		1	Big Brother World
		1	Golden Cards
		1	Halloween I
		1	Pantanal
Máquina 7	256 MB	2	Halloween II
Máquina 8	512 MB	2	Bucaneiros
		2	Campeonato Brasileiro 2007
		2	Frutinha
		1	Golden Cards
		1	Halloween I
		2	Halloween II
		2	Sexy ++

O jogo “Halloween II” foi encontrado em 62% dos equipamentos analisados, seguido pelo jogo “Golden Cards”, com 50%, “Halloween I” e “Halloween Mais Forte” com 37% e os jogos “Bucaneiros”, “Pantanal” e “Sexy ++”, presentes em 25% deles. Os demais jogos, encontrados apenas 12% das máquinas.

Porém, cabe salientar que esta verificação preliminar de similaridade foi realizada tão somente com base nas informações apresentadas pelos equipamentos em sua interface quando em funcionamento.

Assim, visando atender às duas primeiras etapas do modelo de Reengenharia de Software proposto pelo NIST (Peter & Pedrycz, 2001), que possui características semelhantes ao modelo apresentado por Pressman (2006), adotou-se a seguinte metodologia de exames:

a) Para se estabelecer uma concepção formal da aplicação, correspondente à primeira etapa do modelo adotado, foi realizada uma análise da interface do programa com base na usabilidade das telas da aplicação principal e aplicações auxiliares;

b) A segunda etapa, encarregada da extração e análise de informações, foi alcançada através do relacionamento da concepção formal, obtida na etapa anterior, com a análise do ambiente operacional do equipamento.

c) Ainda referente à segunda etapa, a obtenção dos "dados de domínio", que representam as informações de entrada e saída, foi subsidiada pela análise dos itens de *hardware* e pela análise do comportamento da aplicação, com base na sua estrutura interna, diante da interação sofrida. Os "dados de controle", referentes aos comportamentos de decisão e iteração adotados pela aplicação, por sua vez, foram obtidos através da análise do comportamento operacional baseado nos parâmetros de configuração existentes na base de dados.

Durante a análise, adotou-se a abordagem proposta por Vênere (2009, p.118-143), dando prioridade para os elementos mais facilmente identificados, como cadeias de caracteres, por exemplo. Apesar de aquela metodologia direcionar-se a programas de computador compilados em arquiteturas de 32-bits, não se aplicando na totalidade ao presente trabalho, a sugestão de busca de funções foi empregada com propósito diverso daquele apresentado. Utilizou-se tal informação para subsidiar a discussão sobre as similaridades existentes entre programas aparentemente distintos (Capítulo 7).

6.1. USABILIDADE DO SOFTWARE

Nos estabelecimentos comerciais, quando descobertas, as máquinas geralmente estão ligadas e com um determinado jogo em execução. Porém, durante a inicialização de tais

equipamentos é possível verificar que cada máquina exibe uma tela inicial para que o apostador escolha um dos jogos. Assim, cada botão do teclado (Figura 6.1) corresponde a um dos jogos disponibilizados pelo equipamento.

Caso nenhuma opção seja escolhida pelo apostador, a aplicação faz um sorteio aleatório e inicializa um dos jogos presentes em seu dispositivo principal de armazenamento.

Uma vez escolhido um dos jogos exibidos na tela inicial (Figura 6.2), ele é aberto em tela cheia exibindo em seu pano de fundo o motivo que dá nome ao jogo. No terço superior da tela são exibidas as informações referentes ao "crédito" e ao "prêmio". O "crédito" refere-se ao saldo que o apostador tem naquela máquina específica, resultado dos valores inseridos através do noteiro e/ou acumulados através de prêmios já recebido anteriormente. O "prêmio" a ser eventualmente pago é calculado com base na aposta que foi realizada, caso haja um resultado favorável ao apostador. O valor do prêmio tende ao crescimento a cada resultado desfavorável. É possível multiplicar o valor da "aposta" (consequentemente do prêmio), através do produto da quantia de combinações (linhas) escolhidas, por um fator multiplicador entre 1 e 10, escolhido através do botão "aposta" e cujo valor será deduzido do crédito do apostador.

Figura 6.2. Tela para escolha de jogos oferecidos pela Máquina 8 da Tabela 6.1.

No terço inferior da tela principal (Figura 6.3), o jogo exibe o valor “acumulado” (*jackpot*) pela máquina e que corresponde à premiação especial paga em caso de resultado compatível com uma sequência pré-estabelecida. Quanto maior o valor acumulado por uma máquina, maior é o apelo para que os apostadores a escolham. Um determinado jogo, conforme a sua versão, pode apresentar um ou mais acumulados.

O terço médio da tela, por sua vez, apresenta a área de exibição das figuras sorteadas, a qual é composta por 05 (cinco) colunas, contendo 03 (três) figuras cada, representando uma matriz de 3 x 5, que totaliza uma quantia de 15 (quinze) quadrantes, onde são distribuídas as 10 (dez) figuras distintas que compõem o jogo, conforme o sorteio realizado.

Figura 6.3. Tela principal do jogo “Halloween I” com a opção “linha 9” ativada.

À direita e à esquerda da matriz (Figura 6.3), são exibidos valores que variam de 1 a 20, em máquinas cuja opção máxima de jogada é a de “20 linhas”, 1 a 25 em máquinas que oferecem a opção “25 linhas” ou 1 a 30 naqueles equipamentos que permitem a escolha da opção “30 linhas”. Tais valores correspondem à quantia máxima de linhas (jogadas) suportadas por cada equipamento. Os valores numéricos, referentes a cada linha, são exibidos na cor branca em uma elipse com fundo de cor preta enquanto não selecionados.

Ao selecionar uma opção de linhas através do teclado, os valores correspondentes a ela passam a ter uma cor de fundo distinta das demais. Isso permite que o apostador tenha uma clara percepção sobre quais combinações serão premiadas.

As linhas são pré-estabelecidas pelo desenvolvedor da aplicação e são sempre formadas por 05 (cinco) imagens dispostas sequencialmente na horizontal ou diagonal. Assim, quando escolhida a opção "linha 1", a máquina demonstra, através da aplicação de uma linha de cor sólida, quais são as posições no quadrante das figuras cujas combinações serão computadas para pagamento da premiação. Na Figura 6.3 a "linha 1" é composta pelo segundo elemento de cada coluna.

Quanto mais linhas o apostador escolher, maior será o valor deduzido do seu crédito e disponibilizado na forma de aposta na rodada corrente, já que os créditos são deduzidos na proporção de 1 (um) para cada linha escolhida, na modalidade de aposta simples ou 10 (dez) para um, na modalidade de aposta máxima. Assim, cada aposta em "20 linhas", realizada na modalidade de aposta máxima, comprometerá 200 (duzentos) créditos do apostador.

$$Aposta = n(Linhas\ Escolhidas)\ x\ Fator\ de\ Aposta \tag{6.1}$$

Até que sejam inseridos créditos, o equipamento permanece no modo de demonstração, exibindo uma a uma as linhas de premiação predefinidas ou a tabela referente ao valor pago para as ocorrências de repetição nos quinze quadrantes de cada uma das dez figuras exibidas, sem permitir qualquer forma de interação, senão através do noteiro.

Depois de inseridos créditos o apostador poderá:

a) escolher em quantas linhas deseja apostar, pressionando qualquer um dos botões referentes a linhas;

b) exibir a tabela de premiação através do botão "tabela";

c) realizar uma jogada automática, com escolha de linhas e valores de aposta por parte da própria máquina, através do botão "auto";

d) optar pela aposta máxima permitida pelo equipamento, configurando automaticamente o valor máximo de linhas e a modalidade máxima de aposta, através do botão "aposta máxima";

e) escolher um valor entre 1 e 10 para a modalidade de aposta, que será

multiplicado pela quantia de linhas escolhidas, através do botão “aposta”;

f) realizar o sorteio depois de configuradas as suas opções de aposta, através do botão “jogar”;

g) encerrar sua sessão na máquina, resgatando os créditos existentes, através de botão afixado na lateral do gabinete, caso esta opção seja oferecida pelo equipamento.

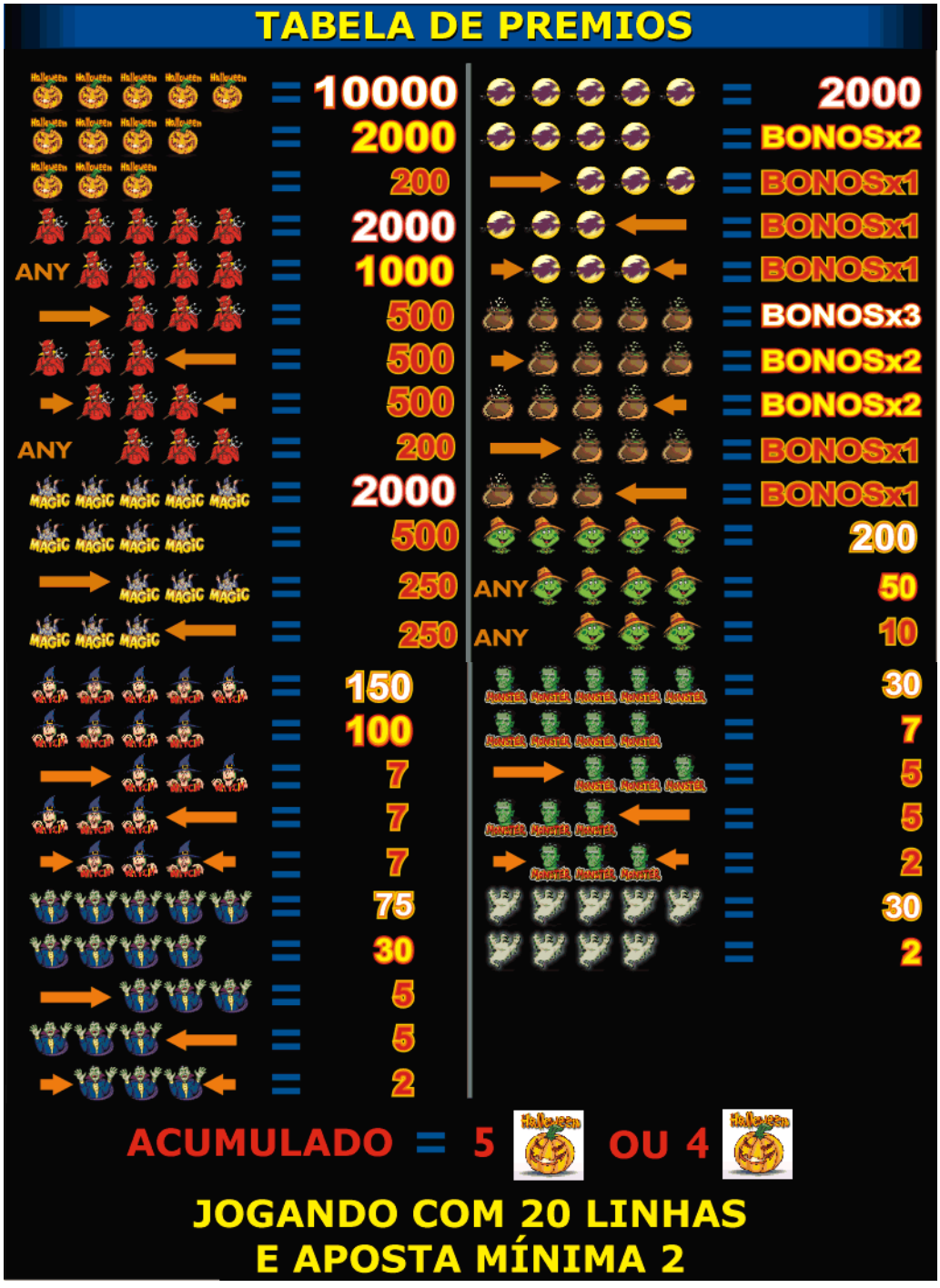

Figura 6.4. Tabela de combinações de premiação do jogo “Halloween I”.

As dez imagens que compõem as opções de sorteio oferecem 34 (trinta e quatro) combinações de prêmios diretos (Figuras 6.4), que variam de 2 a 10.000 créditos, além de 09 (nove) combinações de prêmios pagos a título de bonificação, mediante desempenho realizado em uma tela secundária (Figura 6.5). O resultado pode ser multiplicado por um fator de 1 a 3, conforme a sequência de figuras que levou à bonificação.

Em tese, o pagamento do valor acumulado (prêmio especial) pela máquina será realizado quando o sorteio resultar em uma das duas primeiras sequências da tabela de premiação, desde que a máquina esteja operando na opção de "20 linhas", com fator de aposta igual ou superior a 2.

Obtendo um dos resultados que concedem bônus (Figura 6.4), o apostador é conduzido a uma tela (Figura 6.5) composta por 04 (quatro) colunas e 03 (três) linhas, representando uma matriz de 3 x 4, onde são exibidas 12 (doze) figuras idênticas, porém divergente daquelas exibidas na tela principal, que escondem 11 (onze) figuras com valores diferentes de bônus e somente uma figura com a mensagem "fim de bônus".

Figura 6.5. Tela de bonificação.

Ao escolher uma opção que esconde um valor numérico, este é computado para pagamento da bonificação ao final da etapa e o apostador tem a chance de escolher um novo quadrante. A bonificação é interrompida assim que o quadrante que exibe a imagem de

"fim de bônus" é encontrada (Figura 6.5).

6.2. ITENS DE HARDWARE

Apesar de variar em sua forma externa, que geralmente consiste de uma caixa em madeira pintada na cor preta, evoluindo ao longo dos anos para um formato cada vez mais compacto (Figura 6.1), as máquinas caça-níquel atuais são dotadas de uma placa-mãe da plataforma *personal computer*, a qual abriga um conjunto processador da arquitetura de 32-bits, apresentando, na maioria dos casos, baixa capacidade de memória RAM e reduzida capacidade de armazenamento.

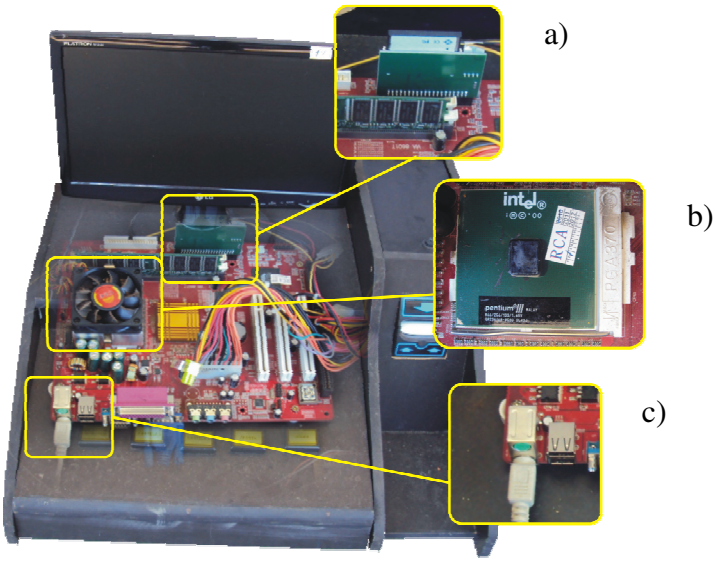

Figura 6.6. Composição gráfica ilustrando o interior dos equipamentos.

6.2.1. Dispositivo de Armazenamento

A área de realce identificada em "a" (Figura 6.6) ilustra o cartão de memória usado como único dispositivo de armazenamento permanente presente no equipamento analisado, sendo responsável por acondicionar tanto a aplicação principal quanto o sistema operacional e os programas auxiliares, destinados à preparação do ambiente.

Conectado à interface IDE (*Integrated Drive Electronics*) da placa-mãe, o cartão de memória sugere que tais aplicações não necessitam de alta capacidade de memória RAM (*Random Access Memory*) ou recursos de vídeo, pois todo o seu conteúdo poderia ser facilmente mantido na memória de execução dos equipamentos comercializados atualmente.

6.2.2. Conjunto Microprocessador

Durante a análise do *hardware*, observou-se que todos os microprocessadores pertencem à plataforma PC (*personal computer*), com predominância dos modelos Pentium III do fabricante Intel[7] e K6 II do fabricante AMD[8], conforme destacado pela alínea "b" da Figura 6.6, embora a aplicação tenha se mostrado compatível com qualquer outro processador da família 8086 e posteriores, com capacidade superior ou inferior à dos itens analisados, conforme exames realizados.

6.2.3. Teclado

O teclado das máquinas caça-níquel tipo "Halloween" é composto por um conjunto de dez botões retangulares ou circulares dispostos em duas linhas conforme a sua funcionalidade. Na linha superior, os botões são comumente rotulados como "Linha 1", "Linhas 5", "Linha 9", "Linha 15" e "Linha 20" por corresponderem às "modalidades de aposta" oferecidas pela aplicação. A linha inferior apresenta botões com rótulos tais como "Tabela" e "Aposta", que possuem a funcionalidade de exibição da tabela de premiação (Figura 6.4) e alteração do "fator de aposta" utilizado no cálculo (6.1) do valor total da aposta. O botão "Aposta máxima" configura, com os maiores valores disponíveis, os itens "modalidade de aposta" e "fator de aposta" e dispara o mecanismo de sorteio. O sorteio também é disparado através dos botões "Auto" e "Aposta", com a diferença que o primeiro deles coloca o equipamento em um modo iterativo de sorteio, enquanto que o segundo dispara aquela funcionalidade uma única vez.

Algumas máquinas contam ainda com um botão auxiliar, geralmente no formato circular com o rótulo "Pagar" ou sem qualquer descrição, responsável por "encerrar a sessão" do apostador na máquina, contabilizando o valor que deverá ser resgatado junto ao estabelecimento.

A conexão do equipamento com o seu teclado dá-se de forma direta ou indireta. Na forma direta, é utilizada uma placa de circuito impresso, dotada de porta paralela, que é conectada diretamente aos barramentos ISA ou PCI. A conexão indireta, por sua vez, utiliza uma placa emuladora de teclado, encarregada de converter o sinal paralelo em serial e entregá-lo no barramento correspondente com o auxílio de conectores DIN 5 ou PS/2, conforme

[7] www.intel.com
[8] www.amd.com

área de realce ilustrada em “c” (Figura 6.6).

Na Figura 6.7 é possível verificar que os equipamentos que utilizam a forma de conexão direta do teclado, não dispõem de dispositivos de armazenamento conectado ao barramento IDE, assim como não apresentam qualquer dispositivo conectado ao barramento serial pela porta PS/2 ou DIN 5, uma vez que a placa de circuito impresso, por possuir memória interna, assume as duas funções: armazenamento do jogo e conexão com o teclado.

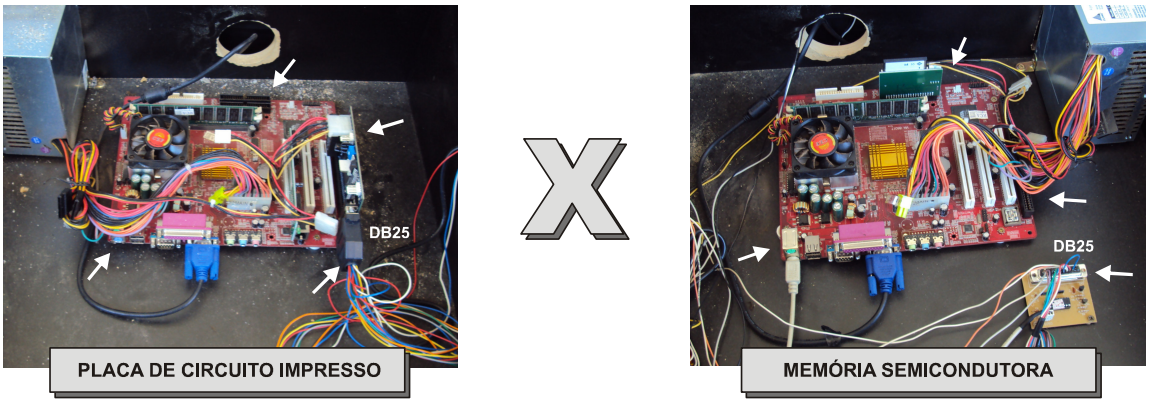

Figura 6.7. Diferenças entre as máquinas caça-níquel com jogo gravado em placa de circuito impresso ou em memória semicondutora.

Por sua vez, as máquinas que se utilizam de memória semicondutora conectada ao barramento IDE adotam a forma de conexão indireta. As placas emuladoras de teclado instaladas em máquinas caça-níquel recebem, através da porta paralela, informações enviadas pelo conjunto de botões que compõem o teclado do equipamento, assim como pelo noteiro (dispositivo responsável por fazer a leitura das cédulas de moeda corrente inseridas pelo apostador). Com fabricação também em território brasileiro (Amwatech, 2011a e Amwatech, 2011b), os emuladores de teclado adotam um esquema particular de conexão, conforme ilustrado no apêndice A.

Os botões que compõem o teclado da máquina, por estarem conectados a pinos distintos do conector DB25, geram códigos também distintos, que serão convertidos pela placa emuladora de teclado ou pela placa de circuito impresso em valores ASCII conforme Tabela 6.2 a seguir.

Tabela 6.2. Correspondência dos pinos da porta paralela com os valores ASCII gerados.

Funcionalidade	Tecla	ASCII	Hexa	Pino 1	Pino 2
Linha 1	Y	89	59	5	13
Linha 5	J	74	4A	5	12
Linha 9	H	72	48	5	11
Linha 15	M	77	4D	5	10
Linha 20	N	78	4E	5	9
Tabela	T	84	54	4	13
Auto	V	86	56	4	12
Aposta Máxima	F	70	46	4	11
Aposta	G	71	47	4	10
Jogar	B	66	42	4	8
Encerrar jogo	R	82	52	5	6
Noteiro	P	80	50	24	25

A coluna “Funcionalidade” da Tabela 6.2 refere-se às opções de interação disponibilizadas ao apostador. Por sua vez, as colunas “Pino 1” e “Pino 2” indicam a quais pinos do conector DB25 cada botão é conectado (Figuras A.1 e A.2) para gerar os valores apresentados na coluna “ASCII”. A coluna “Hexa” indica o valor correspondente em base hexadecimal de cada código gerado pela combinação dos pinos do conector DB25. Finalmente, a coluna “Tecla” faz a correspondência entre os botões apresentados pelo equipamento e um teclado convencional.

6.2.4. Noteiro

Os noteiros, por sua vez, são produzidos por empresas estrangeiras (ICT, 2011a) e importados para montagem das máquinas em território brasileiro. Apesar da origem tailandesa, os noteiros presentes na maioria das máquinas examinadas, reconhecem cédulas de moeda corrente nacional. Tal funcionalidade deve-se ao fato de que esse item de *hardware* possibilita, conforme o fabricante e modelo, a calibração de até 8 (oito) padrões diferentes de cédulas de papel moeda, abrangendo quase que a totalidade das cédulas em circulação, podendo ser adequado à moeda de qualquer País. A personalização (calibração) desse dispositivo de leitura apresenta um baixo grau de dificuldade, apesar de necessitarem de um equipamento específico a este fim, e são amplamente divulgados (ICT, 2011b, Youtube, 2010a e Youtube, 2010b).

Figura 6.8. Modelo de noteiro P70 (ICT, 2011a).

O funcionamento desse dispositivo consiste no envio de uma quantia *n* de pulsos, previamente estipulados para cada padrão calibrado. Os pulsos são recebidos pela placa emuladora de teclado (ICT, 2011c), como se a tecla "P" houvesse sido pressionada *n* vezes no teclado convencional.

Assim, conforme estabelecido pela correspondência de conexão dos pinos 24 e 25 da porta paralela (Tabela 6.2), cada padrão (cédula de papel moeda) calibrado enviará a quantia de pulsos consecutivos correspondentes. Cada pulso será convertido pela placa emuladora em uma sequência de bytes com valor 50.

6.3. AMBIENTE OPERACIONAL DO JOGO

A preparação do ambiente operacional do jogo foi caracterizada através da identificação e classificação dos arquivos presentes no dispositivo principal de armazenamento, separando aqueles que são de uso geral pelo sistema daqueles de uso exclusivo e que possuem influência direta na aplicação. Além disso, dentre os arquivos listados, foram identificados aqueles protegidos por senha ou dissimulados com o objetivo de dificultar o seu reconhecimento pelo sistema operacional e ferramentas de análise forense.

6.3.1. Identificação dos arquivos

A análise do interior dos dispositivos de armazenamento revelou que todos os equipamentos possuem em comum um conjunto de arquivos armazenados na raiz da única partição (FAT) presente no dispositivo de armazenamento (vide seção 6.2.1). Ainda na raiz

da partição há um diretório denominado “dos” e outro denominado “data”, destinados ao armazenamento de arquivos da aplicação e da base de dados, respectivamente.

Tabela 6.3. Conteúdo comum entre os dispositivos de armazenamento das máquinas.

Nome	Tipo	Funcionalidade	Data
data	Diretório	Armazena base de dados	-
dos	Diretório	Armazena arquivos da aplicação	-
autoexec.bat	Arquivo	Inicialização do ambiente	17/03/2007
jogos.txt	Arquivo	Relação de jogos da máquina	14/12/2006
config.sys	Arquivo	Configuração do sistema	29/11/2006
command.com	Arquivo	Prompt de comandos DOS	31/05/1994
drvspace.bin	Arquivo	Recurso do DOS	31/05/1994
io.sys	Arquivo	Recurso do DOS	31/05/1994
msdos.sys	Arquivo	Recurso do DOS	31/05/1994
null	Arquivo	Arquivo vazio	16/03/2007

Durante a avaliação dos arquivos presentes na raiz da partição (Tabela 6.3), foi percebido um lapso temporal aproximado de 12 (doze) anos existente entre os arquivos “drvspace.bin”, “io.sys”, “msdos.sys” e “command.com” e os demais arquivos, sugerindo a mera utilização de arquivos produzidos por terceiros. Enquanto os três primeiros correspondem à preparação de recursos de *hardware*, o último representa a versão 6.22 do prompt de comandos do sistema operacional MS-DOS, conforme evidenciado pelas cadeias de caracteres extraídas através da ferramenta IDA Pro.

O arquivo “jogos.txt” mostrou ser apenas um arquivo de texto cujo conteúdo consiste de uma relação de nomes de supostos jogos que não necessariamente coincidem com aqueles efetivamente oferecidos pela máquina, não oferecendo qualquer subsídio aparente para o funcionamento do sistema.

Ao examinar o arquivo “autoexec.bat”, verificou-se que durante a inicialização da máquina, além de configurar a variável de ambiente “path” para o diretório “c:\dos”, levanta-se a hipótese de criação de uma unidade virtual representada pela letra “d”, onde é chamada posteriormente a execução do aplicativo “programa.exe”, caso este exista no local especificado.

Responsável por preparar o sistema para execução dos aplicativos, o arquivo “config.sys” configura parâmetros de recursos armazenados no diretório “dos”, bem como variáveis de ambiente utilizadas pela aplicação, enquanto que o arquivo “null”, que não possui conteúdo algum, não apresenta, em um primeiro momento, relação significativa com o objeto de estudo.

Além dos arquivos que as máquinas analisadas têm em comum, cada equipamento apresenta um ou mais arquivos com nome formado de forma aparentemente aleatória por 08 (oito) caracteres em caixa alta, com tamanho variando entre 12 e 21 KB, cujo conteúdo binário possui cadeias de caracteres que fazem alusão à tela de seleção de jogos (Figura 6.3).

Entretanto, tais arquivos, quando submetidos ao aplicativo TrID[9], com base na sua assinatura de cabeçalho, apresentam um probabilidade de 100% (conforme resultado da ferramenta) de se tratar de uma trilha de áudio no formato "mp3", cuja representação gráfica demonstrou possuir duração de aproximadamente 70 milissegundos.

Sabendo que um arquivo de áudio com duração inferior a um décimo de segundo é insuficiente para reproduzir qualquer ruído que possa ser utilizado pela aplicação, somando ao fato de que esses arquivos apresentam cadeias de caracteres que fazem alusão ao carregamento da aplicação principal, levanta-se a hipótese deles utilizarem mecanismos de disfarce de informações, como esteganografia, para ofuscar dados que serão providos à aplicação quando da exibição da tela de seleção de jogos. Porém, na sequência da pesquisa identificou-se que tais arquivos estão vinculados ao mecanismo de proteção contra cópia não autorizada desenvolvido pelo fabricante da aplicação e por não influírem na jogabilidade do ambiente, não tiveram sua análise aprofundada.

Tabela 6.4. Conteúdo do diretório "data" comum entre as máquinas.

Nome	Tipo	Funcionalidade	Data
base.dbf	Arquivo	Base de dados dBase	-
macid.[01-10]	Arquivo	Arquivos de codificação da base	26/10/2006
macid.11	Arquivo	Sequência numérica variável	Variável
macid.bin	Arquivo	Cadeia de caracteres única	15/09/2006
pago.pcx	Arquivo	Imagem	18/07/2006

Ao verificar o conteúdo dos arquivos de nome "macid", localizado no diretório "data" (Tabela 6.4), verificou-se que, com exceção daquele com a extensão "11" (que possui uma sequência numérica de comprimento variável e composição distinta em cada máquina), os demais arquivos (com extensão numérica entre 01 e 10 ou com a extensão "bin"), possuem comprimento de 20 (vinte) e 24 (vinte e quatro) caracteres (respectivamente). O conteúdo desses arquivos, formado por símbolos da tabela ASCII, é idêntico em todos os equipamentos analisados e supostamente correspondem ao sistema de proteção contra

[9] trid.softonic.com.br

cópia não autorizada constatado nas máquinas durante a análise.

O arquivo “base.dbf”, além de apresentar nome que sugere uma base de dados, possui assinatura compatível com a base de dados em arquivo dBase, conforme constatado pelo aplicativo TrID com probabilidade de convergência de 99,7%. Ao ser editado, tal arquivo revelou uma listagem de campos destinados à configuração do ambiente do aplicativo executado.

Contudo, ao examinar os campos presentes no arquivo “base.dbf” verificou-se que alguns deles não possuíam valor armazenado, sugerindo serem utilizados somente quando da execução da aplicação, enquanto que o restante apresentava conteúdo ininteligíveis, devido a alguma espécie de codificação utilizada, não identificada de imediato. Apenas os campos que traziam valores em ponto flutuante, referentes aos índices de acumulação da máquina, apresentaram-se legíveis durante a análise *off-line*.

Tabela 6.5. Conteúdo do diretório “dos” comum entre as máquinas.

Nome	Tipo	Funcionalidade	Data
pkunzip.exe	Arquivo	Utilitário de descompressão	01/02/1993
cntdwn16.fnt	Arquivo	Fonte tipográfica para DOS	01/02/1994
ocr_16.fnt	Arquivo	Fonte tipográfica para DOS	01/02/1994
thin_16.fnt	Arquivo	Fonte tipográfica para DOS	01/02/1994
filefix.exe	Arquivo	Utilitário de recuperação de DBF	28/02/1994
filefix.hlp	Arquivo	Utilitário de recuperação de DBF	28/02/1994
filefix.ico	Arquivo	Utilitário de recuperação de DBF	28/02/1994
nlib200.rtl	Arquivo	Utilitário Norton para DOS	28/02/1994
ansi.sys	Arquivo	Recurso do DOS	31/05/1994
append.exe	Arquivo	Recurso do DOS	31/05/1994
attrib.exe	Arquivo	Recurso do DOS	31/05/1994
chkdsk.exe	Arquivo	Recurso do DOS	31/05/1994
chkstate.sys	Arquivo	Recurso do DOS	31/05/1994
choice.com	Arquivo	Recurso do DOS	31/05/1994
command.com	Arquivo	Recurso do DOS	31/05/1994
config.exe	Arquivo	Recurso do DOS	31/05/1994
country.sys	Arquivo	Recurso do DOS	31/05/1994
dblwin.hlp	Arquivo	Recurso do DOS	31/05/1994
debug.exe	Arquivo	Recurso do DOS	31/05/1994
defrag.exe	Arquivo	Recurso do DOS	31/05/1994
defrag.hlp	Arquivo	Recurso do DOS	31/05/1994
deltree.exe	Arquivo	Recurso do DOS	31/05/1994
diskcomp.com	Arquivo	Recurso do DOS	31/05/1994
display.sys	Arquivo	Recurso do DOS	31/05/1994
doshelp.hlp	Arquivo	Recurso do DOS	31/05/1994
doskey.com	Arquivo	Recurso do DOS	31/05/1994
dossetup.ini	Arquivo	Recurso do DOS	31/05/1994

driver.sys	Arquivo	Recurso do DOS	31/05/1994
drvspace.bin	Arquivo	Recurso do DOS	31/05/1994
drvspace.exe	Arquivo	Recurso do DOS	31/05/1994
drvspace.hlp	Arquivo	Recurso do DOS	31/05/1994
drvspace.inf	Arquivo	Recurso do DOS	31/05/1994
drvspace.sys	Arquivo	Recurso do DOS	31/05/1994
drvspace.txt	Arquivo	Recurso do DOS	31/05/1994
edit.com	Arquivo	Recurso do DOS	31/05/1994
ega.cpi	Arquivo	Recurso do DOS	31/05/1994
ega2.cpi	Arquivo	Recurso do DOS	31/05/1994
ega3.cpi	Arquivo	Recurso do DOS	31/05/1994
emm386.exe	Arquivo	Recurso do DOS	31/05/1994
expand.exe	Arquivo	Recurso do DOS	31/05/1994
fasthelp.exe	Arquivo	Recurso do DOS	31/05/1994
fastopen.exe	Arquivo	Recurso do DOS	31/05/1994
fc.exe	Arquivo	Recurso do DOS	31/05/1994
format.com	Arquivo	Recurso do DOS	31/05/1994
graphics.com	Arquivo	Recurso do DOS	31/05/1994
graphics.pro	Arquivo	Recurso do DOS	31/05/1994
help.com	Arquivo	Recurso do DOS	31/05/1994
help.hlp	Arquivo	Recurso do DOS	31/05/1994
himem.sys	Arquivo	Recurso do DOS	31/05/1994
info.txt	Arquivo	Recurso do DOS	31/05/1994
interlnk.exe	Arquivo	Recurso do DOS	31/05/1994
intersvr.exe	Arquivo	Recurso do DOS	31/05/1994
iso.cpi	Arquivo	Recurso do DOS	31/05/1994
keyb.com	Arquivo	Recurso do DOS	31/05/1994
keyboard.sys	Arquivo	Recurso do DOS	31/05/1994
keybrd2.sys	Arquivo	Recurso do DOS	31/05/1994
label.exe	Arquivo	Recurso do DOS	31/05/1994
leiame.txt	Arquivo	Recurso do DOS	31/05/1994
loadfix.com	Arquivo	Recurso do DOS	31/05/1994
mem.exe	Arquivo	Recurso do DOS	31/05/1994
memmaker.exe	Arquivo	Recurso do DOS	31/05/1994
memmaker.hlp	Arquivo	Recurso do DOS	31/05/1994
memmaker.inf	Arquivo	Recurso do DOS	31/05/1994
mode.com	Arquivo	Recurso do DOS	31/05/1994
monoumb.386	Arquivo	Recurso do DOS	31/05/1994
more.com	Arquivo	Recurso do DOS	31/05/1994
move.exe	Arquivo	Recurso do DOS	31/05/1994
mscdex.exe	Arquivo	Recurso do DOS	31/05/1994
msd.exe	Arquivo	Recurso do DOS	31/05/1994
nlsfunc.exe	Arquivo	Recurso do DOS	31/05/1994
power.exe	Arquivo	Recurso do DOS	31/05/1994
print.exe	Arquivo	Recurso do DOS	31/05/1994
qbasic.exe	Arquivo	Recurso do DOS	31/05/1994
qbasic.hlp	Arquivo	Recurso do DOS	31/05/1994
ramdrive.sys	Arquivo	Recurso do DOS	31/05/1994

redes.txt	Arquivo	Recurso do DOS	31/05/1994
replace.exe	Arquivo	Recurso do DOS	31/05/1994
restore.exe	Arquivo	Recurso do DOS	31/05/1994
scandisk.exe	Arquivo	Recurso do DOS	31/05/1994
scandisk.ini	Arquivo	Recurso do DOS	31/05/1994
setver.exe	Arquivo	Recurso do DOS	31/05/1994
share.exe	Arquivo	Recurso do DOS	31/05/1994
sizer.exe	Arquivo	Recurso do DOS	31/05/1994
smartdrv.exe	Arquivo	Recurso do DOS	31/05/1994
sort.exe	Arquivo	Recurso do DOS	31/05/1994
subst.exe	Arquivo	Recurso do DOS	31/05/1994
sys.com	Arquivo	Recurso do DOS	31/05/1994
tree.com	Arquivo	Recurso do DOS	31/05/1994
undelete.exe	Arquivo	Recurso do DOS	31/05/1994
unformat.com	Arquivo	Recurso do DOS	31/05/1994
vfintd.386	Arquivo	Recurso do DOS	31/05/1994
vsafe.com	Arquivo	Recurso do DOS	31/05/1994
xcopy.exe	Arquivo	Recurso do DOS	31/05/1994
symcfg.bin	Arquivo	Utilitário Norton para DOS	28/03/2001
edit.hlp	Arquivo	Utilitário binário renomeado	18/09/2006
backup.bat	Arquivo	Utilitário para backup	20/10/2006
restaura.bat	Arquivo	Utilitário para restauração de backup	20/10/2006
reboot.com	Arquivo	Força a reinicialização do sistema	13/11/2006
dprot.exe	Arquivo	Utilitário para proteção de cópia	03/03/2007
prot.exe	Arquivo	Utilitário para proteção de cópia	10/03/2007
idhdpro.exe	Arquivo	Utilitário para proteção de cópia	16/03/2007
logomenu.gif	Arquivo	Imagem GIF	21/03/2007
menu.exe	Arquivo	Aplicativo que carrega os jogos	30/03/2007
fdisk.exe	Arquivo	Arquivo texto renomeado	23/07/2007
find.exe	Arquivo	Arquivo texto renomeado	23/07/2007
jogos.bin	Arquivo	Base de dados dBase renomeada	23/07/2007

Da mesma forma, ao examinar o conteúdo do diretório “dos” (Tabela 6.5) de todas as máquinas, constatou-se que elas apresentavam 106 (cento e seis) arquivos em comum, dos quais 85 (oitenta e cinco) deles, datados de 31/05/1994, correspondem a recursos do MS-DOS, não sendo examinados minuciosamente por não apresentarem relevância com a problemática em estudo.

Dos 21 (vinte e um) arquivos restantes, verificou-se ainda que aqueles de nome “nlib200.rtl” e “symcfg.bin” pertencem ao utilitário Norton para ambiente DOS; aqueles com extensão “fnt” correspondem à fontes tipográficas utilizadas em ambiente DOS; os 03 (três) arquivos de nome “filefix” pertencem a um utilitário de reparação de arquivos DBF corrompidos e o arquivo “pkunzip.exe” é um utilitário de descompressão de arquivos binários.

Uma vez desconsiderados os arquivos conhecidos, restaram apenas 12 (doze) dos arquivos presentes em todos os equipamentos analisados e que são condizentes com a problemática em estudo. Tais arquivos são os que apresentam data de criação mais recente que os demais, compreendendo um período de cerca de um ano.

Os arquivos “backup.bat” e “restore.bat”, apesar de apresentarem relação com a aplicação em estudo, nada mais são que arquivos de execução de comandos em lote que copiam o conteúdo do diretório “data” para um diretório “bkp” criado dentro do diretório “dos”. Da mesma forma, o arquivo “reboot.com”, simplesmente executa uma interrupção de sistema que força a reinicialização. Diante de tal simplicidade de funcionamento, tais arquivos não necessitam de maiores esclarecimentos.

Os aplicativos principais das máquinas caça-níquel examinadas possuem um sistema de proteção anticópia de vários níveis, onde são testados diversos itens para identificar supostas tentativas de reprodução não autorizada das aplicações. Um dos níveis de segurança contra proteção é a existência do arquivo, cujo nome é composto por oito caracteres dispostos de forma aparentemente aleatória, localizado na raiz do dispositivo de armazenamento. Tal funcionalidade é gerada pelos arquivos “dprot.exe”, “prot.exe” e “idhpro.exe”. Entretanto, como o estudo do mecanismo de proteção contra cópia não apresenta relevância para a problemática em estudo, tais arquivos foram desconsiderados durante as demais etapas da pesquisa.

O arquivo “menu.exe”, ao ser submetido ao aplicativo String (Microsoft, 2012), retornou cadeias de caracteres que indicam ser o aplicativo responsável pela exibição da tela de seleção de jogos (Figura 6.4).

Porém, dentre os arquivos analisados, quatro deles apresentaram-se com extensão divergente do seu cabeçalho ao serem submetidos à análise pela ferramenta TrID. O arquivo “edit.hlp”, cuja extensão faz alusão a um arquivo de ajuda do utilitário de edição do DOS, apresentou-se como arquivo binário, onde as cadeias de caracteres extraídas são compatíveis com aquelas extraídas do arquivo “menu.exe”. Tal semelhança, somada ao fato de o arquivo “edit.hlp” ter data de criação anterior ao “menu.exe”, sugere a possibilidade de o segundo ter sido gerado a partir do primeiro, quando da aplicação do sistema de proteção contra reprodução não autorizada, mencionado anteriormente.

Os arquivos “fdisk.exe” e “find.exe”, apesar de sugerirem tratar de utilitários conhecidos em ambiente DOS, apresentam conteúdo em texto claro, representado por uma sequência

numérica integralmente compatível com o conteúdo do arquivo “macid.11”, constatado no interior do diretório “data”.

O arquivo “jogos.bin”, por sua vez, apresenta assinatura compatível com a base de dados em arquivo dBase, indicando que teve a extensão alterada para impedir a sua pronta identificação.

Além dos arquivos que as máquinas possuem em comum, foram identificados arquivos com a extensão “sys” (Tabela 6.6), apresentando uma grande variação de tamanho entre eles, onde os nomes e a quantia de arquivos apresentados por cada equipamento são compatíveis com os itens exibidos na tela inicial de seleção de jogos. A convergência entre os arquivos armazenados e as opções apresentadas na tela inicial de seleção, indica que cada arquivo corresponde a um dos jogos oferecidos pelo equipamento.

A diversidade de tamanho apresentada pelos arquivos sugere a hipótese de que os jogos são diferentes quanto a sua estrutura interna. Ao todo, em todos os equipamentos analisados, foram encontrados 23 (vinte e três) arquivos com extensão “sys” distintos. Embora a análise preliminar dos equipamentos tenha apresentado uma lista de 18 (dezoito) jogos distintos (Tabela 6.1), foram constatados arquivos “sys” com nomes divergentes mesmo em jogos aparentemente idênticos quando exibidos pela máquina.

Tabela 6.6. Arquivos com extensão “sys” presentes na maquina ilustrada na Figura 6.4.

Nome	Tipo	Funcionalidade	Tamanho	Data
camp2007.sys	Arquivo	Jogo Brasileirão 2007	1.681 KB	11/07/2007
fruta.sys	Arquivo	Jogo Frutinha	1.617 KB	01/07/2007
gcard1.sys	Arquivo	Jogo Golden Card	1.212 KB	01/05/2007
hallo1.sys	Arquivo	Jogo Halloween I	1.793 KB	01/05/2007
hallo2.sys	Arquivo	Jogo Halloween II	1.795 KB	01/05/2007
sexy2.sys	Arquivo	Jogo Super Sexy	3.019 KB	10/05/2007

Porém, ao submeter tais arquivos à ferramenta TrID, verificou-se que, apesar de terem sido renomeados para uma extensão que é costumeiramente utilizada para indicar arquivos de sistema, tratam-se de arquivos com assinatura compatível à de arquivos compactados no formato ZIP. Confirmando o diagnóstico da ferramenta TrID, ao submeter esses mesmos arquivos à ferramenta Strings[10], dentre as cadeias de caracteres retornadas, foi identificada a informação “PKLITE Copr. 1990-92 PKWARE Inc. All Rights Reserved” em todos eles, fazendo alusão ao uso da ferramenta de compressão de código “PKLite” da empresa

[10] technet.microsoft.com/en-us/sysinternals/bb897439

“PKWare”.

A existência do arquivo “pkunzip.exe” no diretório “dos”, que apresenta apenas a capacidade de descompressão do formato ZIP, sugere que tais arquivos são descompactados durante a execução da máquina.

Uma vez revelado que tais arquivos estavam compactados, procedeu-se com a troca da extensão “sys” para “zip”. Ao visualizar seu conteúdo, verificou-se que apesar da divergência de tamanho do arquivo final, cada um dos jogos consiste de uma quantidade variável (em alguns casos) de arquivos na extensão “dat”, com nomes compatíveis entre si, e um único arquivo executável denominado “programa.exe”.

Tabela 6.7. Comparativo de conteúdo dos arquivos com extensão “sys”.

Nome	Executáveis	Tamanho	Total
bigb1ac.sys	programa.exe	273 KB	339
buca2.sys	programa.exe	281 KB	335
buca2ac.sys	programa.exe	281 KB	341
camp2ac.sys	programa.exe	281 KB	314
camp2007.sys	programa.exe	281 KB	321
fruta.sys	programa.exe	281 KB	315
fuga1ac.sys	programa.exe	273 KB	372
fuga2acm.sys	programa.exe	281 KB	376
gcard1.sys	programa.exe	273 KB	333
goldcard.sys	programa.exe	273 KB	333
hallo1.sys	programa.exe	273 KB	335
hallo1ac.sys	programa.exe	273 KB	335
hallo2.sys	programa.exe	281 KB	335
hallo2ac.sys	programa.exe	281 KB	335
hallo2tk.sys	programa.exe	281 KB	331
hallofor.sys	programa.exe	677 KB	335
oro2ac.sys	programa.exe	677 KB	236
pantanal.sys	programa.exe	273 KB	373
sexy2.sys	programa.exe	281 KB	333
sexy2ac.sys	programa.exe	281 KB	373
trago1ac.sys	programa.exe	273 KB	331
vacalo2.sys	programa.exe	281 KB	374
vakalo1.sys	programa.exe	273 KB	331

Apesar de apresentarem o mesmo nome (Tabela 6.7), nos vinte e três jogos encontrados nos equipamentos utilizados no desenvolvimento da pesquisa, foram identificadas três variações de tamanho para o arquivo “programa.exe”, sugerindo que ao invés de cada jogo possuir uma estrutura interna distinta, vários deles compartilham do mesmo arquivo principal.

Contudo, ao tentar a descompressão, verificou-se que todos os arquivos foram compactados utilizando uma chave de segurança, a qual de alguma forma deve ser informada pelo sistema durante a sua execução, já que foi constatado o utilitário de descompressão do formato ZIP presente no diretório “dos”, permitindo inferir que a chave estaria armazenada em algum arquivo.

6.3.2. Arquivos Protegidos por Senha

Por se tratar de aplicativo executado em ambiente DOS, que dificulta a separação das informações de um processo em específico em meio a uma imagem completa extraída da memória, descartou-se a hipótese de execução de ferramentas que recuperassem a senha de descompressão diretamente da memória de execução.

Diante de tal impasse e baseado na premissa de que as senhas de descompressão poderiam estar inseridas em algum arquivo presente no dispositivo de armazenamento, os arquivos de nome “jogos.bin” de cada máquina, que foram reconhecidos pela ferramenta TrID como sendo tabelas dBase, foram renomeados para a extensão “dbf”.

Ao serem abertos, revelaram a existência de uma tabela com as colunas “nomejogo”, “mascara” e “senha”, todas do tipo texto, bem como de uma quantidade de registros variável entre os arquivos “jogos.bin” de cada máquina, porém compatível com a quantidade de itens apresentados pela sua tela inicial de seleção e compatível também com a quantidade de arquivos com extensão “sys” disponíveis em cada equipamento.

Apesar de as informações armazenadas apresentarem-se ininteligíveis em um primeiro momento, também indicando a adoção de alguma espécie de codificação, tal qual ocorrido com o arquivo “base.dbf” constatado no diretório “data”, foi possível constatar que as senhas cadastradas para todos os jogos possui a mesma representação codificada, com comprimento de seis dígitos.

Tabela 6.8. Conteúdo do arquivo “jogos.bin” da máquina ilustrada na Figura 6.2

NOMEJOGO	MASCARA	SENHA
§³´¸Á°±»Ž¸¹	§³´¸œ™¿ÆÁ	²§¹«ª¼
§³´¸Á°±»Ž¸	§³´¸›™¿ÆÁ	²§¹«ª¼
¬µ³¬®¸‹¯®À³	¬©¨º›™¿ÆÁ	²§¹«ª¼
¸»·»Š¾±ÅÇ	¸«¿Á›˜¾ÅÀ	²§¹«ª¼
§¸¨»²¶°µ¿¯¾ £¢£«	¨§´¸›š›£›ÁÈÃ	²§¹«ª¼
«¸¼¼²¸³	«¸¼¼ª˜¾ÅÀ	²§¹«ª¼

Embora com comprimento reduzido, a abordagem de quebra de senha por força bruta, utilizando-se da combinação de todos os caracteres imprimíveis possíveis, seria um tanto quanto dispendiosa, necessitando de um longo período de espera e inviabilizando a sua adoção. Ao invés disso, direcionaram-se os esforços para a tentativa de descoberta de informações que auxiliassem na elucidação do algoritmo utilizado na codificação.

Na tentativa de se encontrar um padrão entre os símbolos exibidos na coluna “mascara” e os nomes dos arquivos com extensão “sys” constatados no mesmo equipamento que abriga o arquivo “jogos.bin”, percebeu-se que não havia uma correspondência direta de símbolos, pois o dígito “2” presente no nome do arquivo “hallo2.sys”, que supostamente é representado pelo símbolo “œ” (código 156), é representado pelo símbolo “›” (código 155) no arquivo “sexy2.sys”.

Porém, assim como o código numérico que representa ambos os símbolos apresenta a diferença de uma unidade, o caractere “2” também se apresenta com diferença de uma unidade em relação ao seu posicionamento no nome do arquivo, já que no arquivo “hallo2.sys” ele é o sexto caractere, enquanto que no arquivo “sexy2.sys” ele é o quinto, revelando um padrão entre o valor numérico do símbolo e o valor numérico do caractere do texto em claro.

Utilizando-se desse raciocínio, procedeu-se com o confronto dos demais caracteres e as suas possíveis correspondências codificadas, sendo possível inferir que o algoritmo de codificação consiste na substituição do caractere original por outro cuja representação numérica seja compatível com o resultado da soma de uma centena com o valor numérico do caractere que se deseja codificar, somado ainda ao valor referente à sua posição na cadeia de caracteres.

$$texto\_cifrado[i] = texto\_claro[i] + 100 + i; \tag{6.2}$$

Tendo elucidado o algoritmo de codificação, realizou-se a tradução das informações apresentadas pela tabela “jogos”, que além de fornecer a senha de descompressão (MARCAR), demonstrou que os valores presentes no campo “mascara” referiam-se aos nomes dos arquivos que contém o jogo compactado, conforme ilustrado pela tabela a seguir.

Tabela 6.9. Tradução do conteúdo do arquivo “jogos.bin” da máquina da Figura 6.4

NOMEJOGO	MASCARA	SENHA
HALLOWEEN II	HALLO2.SYS	MARCAR
HALLOWEEN I	HALLO1.SYS	MARCAR
GOLDEN CARD	GCARD1.SYS	MARCAR
SUPER SEXY	SEXY2.SYS	MARCAR
BRASILEIRAO 2007	CAMP2007.SYS	MARCAR
FRUTINHA	FRUTA.SYS	MARCAR

6.3.3. Arquivos Dissimulados

Uma vez realizada a descompressão dos arquivos com extensão “sys”, utilizou-se novamente a ferramenta TrID pra verificar o real formato dos inúmeros arquivos com extensão “dat” que os jogos trazem compactados junto com o arquivo executável. Foi revelado pela ferramenta que todos correspondem a imagens no formato “pcx” utilizadas para compor o *layout* do jogo em tempo de execução.

Dentre os arquivos listados, aquele de nome “tela1.dat” refere-se ao pano de fundo principal de cada um dos jogos que, apesar de ter tema divergente, possui áreas destinadas à exibição das mesmas informações. Por serem compatíveis, tais imagens podem ser facilmente substituídas entre si.

Com isso, apesar das divergências visuais apresentadas pela sua interface, há a possibilidade de os jogos possuírem a mesma estrutura interna quanto ao seu comportamento, mesmo que os executáveis apresentem tamanhos distintos.

Ao analisar o cabeçalho binário do arquivo “programa.exe”, a ferramenta TrID apontou grande probabilidade de tratar-se de arquivo binário compilado em arquitetura de 16 bits, portanto completamente incompatível com a ferramenta OllyDBG, que suporta somente a arquitetura de 32 bits e parcialmente incompatível com a ferramenta IDA Pro, visto que a maioria das suas funcionalidades aplicam-se somente à arquitetura mais recente.

Assim, o arquivo “programa.exe” foi submetido à ferramenta Strings, onde, dentre outras informações, constatou-se o termo “BLinker”, que remete ao aplicativo “Borland Linker”, sugerindo que a aplicação tenha sido compilada em uma ferramenta proprietária da Borland, o que restringe o universo às ferramentas Turbo Pascal, Delphi e C++ Builder. A hipótese de ter sido compilado através da segunda ferramenta somente aplicar-se-á caso tenha sido realizado em sua primeira versão, já que a ferramenta DeDe, que se propõe a

decompilar aplicativos escritos nas versões 2 a 10 da ferramenta Borland Delphi[11], não o reconheceu como um código compatível com aquelas versões.

6.4. ESTRUTURA INTERNA DO JOGO

Uma vez coletadas as informações disponíveis através da análise *off-line*, referentes ao ambiente operacional das máquinas caça-níquel, procedeu-se com a análise *online* dos arquivos "menu.exe" e "programa.exe". O primeiro está localizado no diretório "dos" (Tabela 6.5) de cada dispositivo de armazenamento; o segundo é resultado da descompressão dos arquivos "sys" (Tabela 6.7) e corresponde aos jogos oferecidos pelo equipamento.

6.4.1. Tela de Seleção de Jogos

Ao depurar um dos exemplares do arquivo "menu.exe" através da ferramenta CodeView[12], verificou-se que durante a exibição da tela inicial de seleção de jogo, a aplicação chama o sistema operacional, através da interrupção 21, movendo o valor 0x01 para AH, que tem como função a captura de um caractere do teclado (Jurgens, 2011). Caso seja pressionado algum dos botões, a representação de hexadecimal da tecla pressionada é armazenada em AL. Caso contrário, depois de um período pré-determinado, a aplicação faz um sorteio de um dos valores válidos.

Além de realizar comparações diretas com os valores apresentados na coluna "Hexa" da Tabela 6.2, compatíveis com os itens de interação oferecidos por cada equipamento, constatou-se também uma comparação com o valor 0x53, que corresponde à tecla "S" do teclado. Em caso de convergência, é chamada a abertura do arquivo "command.com" armazenado no diretório "dos" e o *prompt* de comandos é exibido ao operador do equipamento, apontando para a unidade de disco "c".

Em virtude da preparação do ambiente operacional, realizado por "menu.exe", observou-se que o acesso à *shell* de comandos através do programa apresenta uma segunda unidade de disco, representada pela letra "d". Em um primeiro momento, a unidade virtual é exibida sem conteúdo e é necessário executar manualmente "menu.exe" para voltar à aplicação. Ao selecionar um dos jogos apresentados na tela inicial, ao invés de visualizar a tela principal

[11] www.borland.com
[12] www.nuvisionmiami.com/books/asm/cv/cv41patch.exe

da aplicação, o usuário é novamente conduzido à *shell* de comandos. Porém, desta vez, a unidade virtual apresenta os arquivos descompactados do arquivo "sys" correspondente.

A confirmação da existência da unidade virtual de disco durante a análise *online* comprova a hipótese surgida quando da análise do arquivo "autoexec.bat" (Seçao 6.3.1) constatado na raiz do dispositivo de armazenamento e que verifica a existência do arquivo "programa.exe" na unidade de disco "d".

6.4.2. Aplicativo Principal do Jogo

Durante a análise binária do aplicativo principal do jogo, foram identificados três níveis de segurança para acesso dissimulado a parâmetros de leitura ou de ajuste de comportamento das máquinas caça-níquel.

6.4.2.1. Ambiente de Configuração

Da mesma forma, ao depurar o arquivo "programa.exe" através da ferramenta CodeView, verificou-se que durante a exibição da tela principal do jogo, além de aguardar pelo código das teclas listadas na coluna "Hexa" da Tabela 6.2, é realizada uma comparação do conteúdo do registrador AL com o valor hexadecimal 0x34, que corresponde à tecla "4" do teclado convencional.

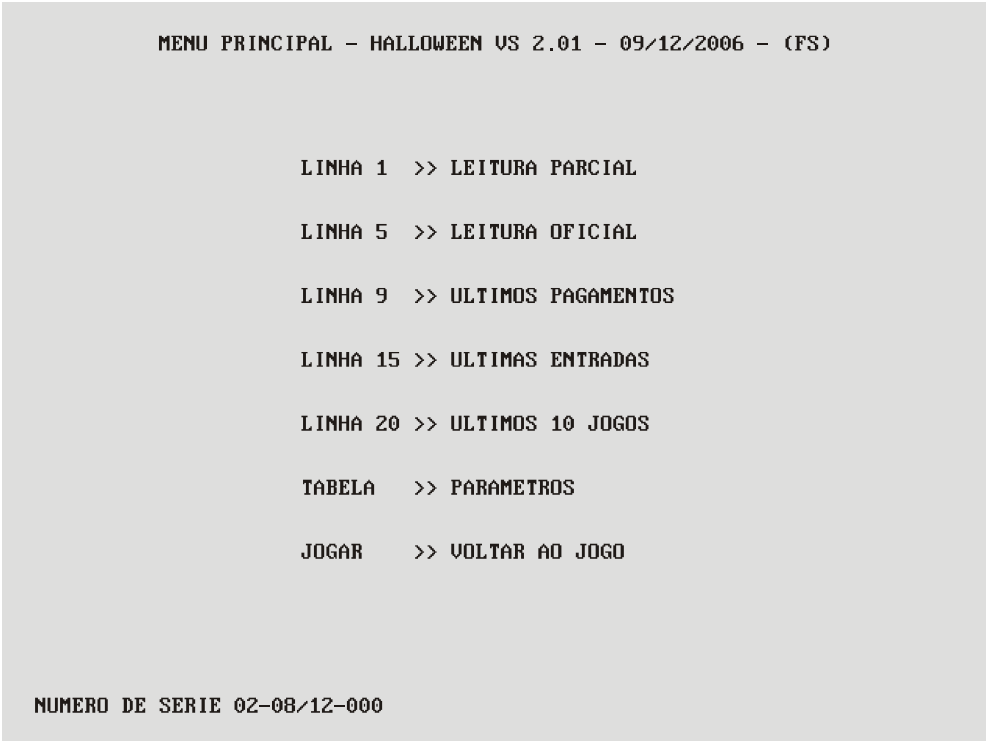

Figura 6.9. Tela de administração do sistema.

Esta *backdoor* leva a uma tela de administração do sistema (Figura 6.9) com opções de leitura estatística de entradas e saídas e configuração de parâmetros do jogo. Esta funcionalidade é exibida somente caso o apostador ainda não tenha iniciado uma sessão com a inserção de créditos.

Dentre as opções apresentadas pela tela de configuração, a primeira delas é a de "leitura parcial", acessível através da tecla "Y" (vide mapeamento de teclas na Tabela 6.2), que exibe as estatísticas da máquina quanto aos valores totais de "entradas", "pagamentos", "apostas" e "prêmios" (Figura 6.10). É possível ainda obter o percentual de prêmios pagos e um balanço que se refere ao lucro da máquina, que corresponde ao resultado da subtração dos "pagamentos" pelo total de "entradas". O percentual de prêmios pagos, por sua vez, é obtido através da proporção do número de prêmios pagos em relação ao número de apostas.

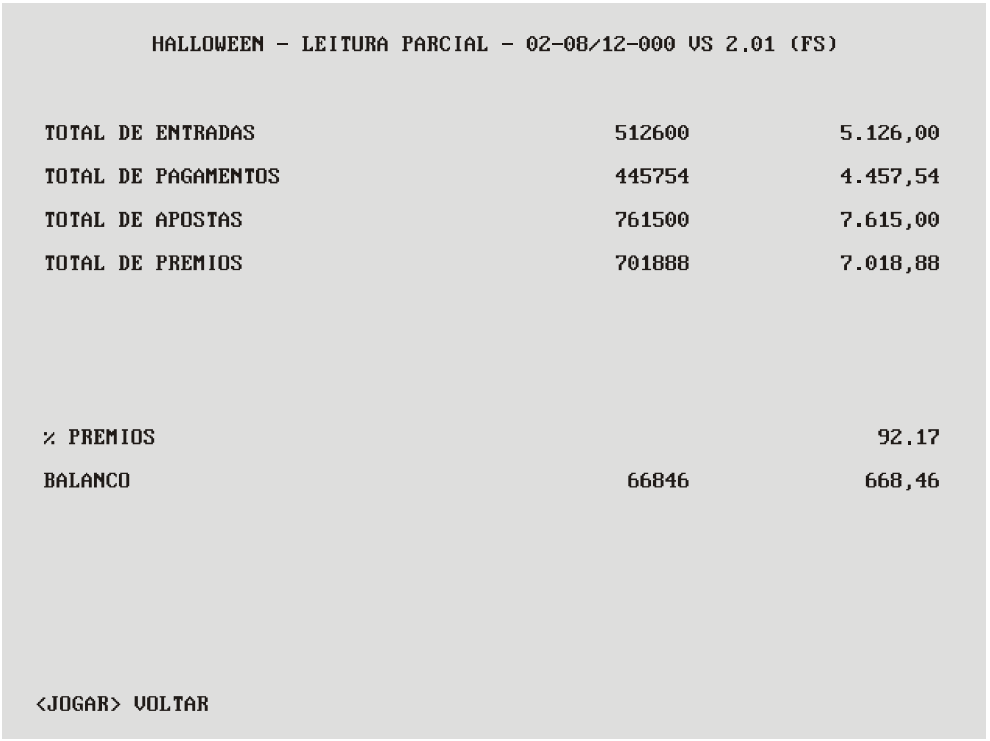

Figura 6.10. Tela de leitura parcial.

6.4.2.2. Primeiro Nível de Segurança

Ao pressionar a tecla "J", referente à opção denominada "leitura oficial", tem-se a impressão de que a aplicação não executa qualquer ação. Porém, ao depurar seu código binário através da ferramenta CodeView, verificou-se que ela aguarda o envio de um valor

através do teclado. Este valor, por sua vez, é confrontado com uma cadeia de caracteres armazenada na seção de dados do arquivo executável, que corresponde à senha correta. Essa prática de armazenamento de senha é conhecida como *hardcoded* (Narvaja, 2005). A forma de localização do *hardcoded* no arquivo será abordada posteriormente, visto terem sido encontradas, ao longo da análise, outras seções da aplicação com verificação de senha.

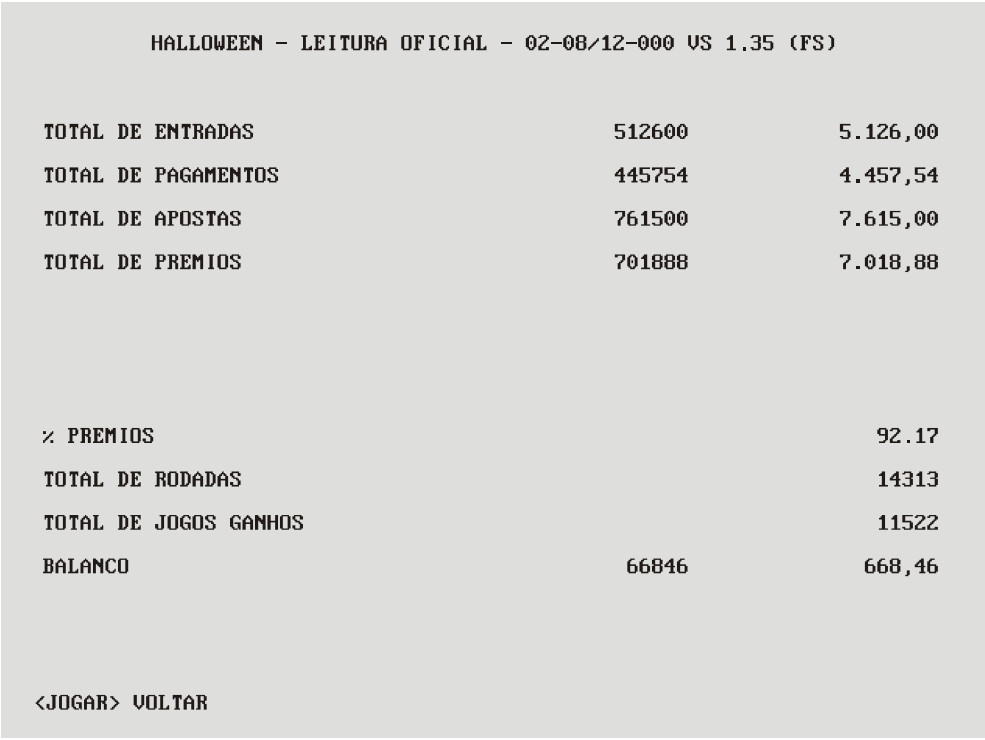

Figura 6.11. Tela de leitura oficial acessível através de senha oculta.

A divergência entre as telas de “leitura parcial” (Figura 6.10) e “leitura oficial” (Figura 6.11) consiste tão somente na possibilidade de reiniciar os contadores da primeira leitura, sem que os valores totais da máquina sejam perdidos. Com isso, durante a análise de uma determinada máquina, a “leitura parcial” pode não refletir os valores totais arrecadados, sendo necessário confrontá-los com os valores da “leitura oficial”. O reinício dos valores da “leitura oficial” é obtido através do pressionamento dos botões referentes à sequência de teclas JHHHJ.

Ao receber o código correspondente ao pressionamento da tecla “H”, a aplicação exibe os dez últimos pagamentos de premiação efetuados pelo equipamento (Figura 6.12). As cinco últimas inserções de crédito efetuadas (Figura 6.13) podem ser obtidas pressionamento da tecla “M”.

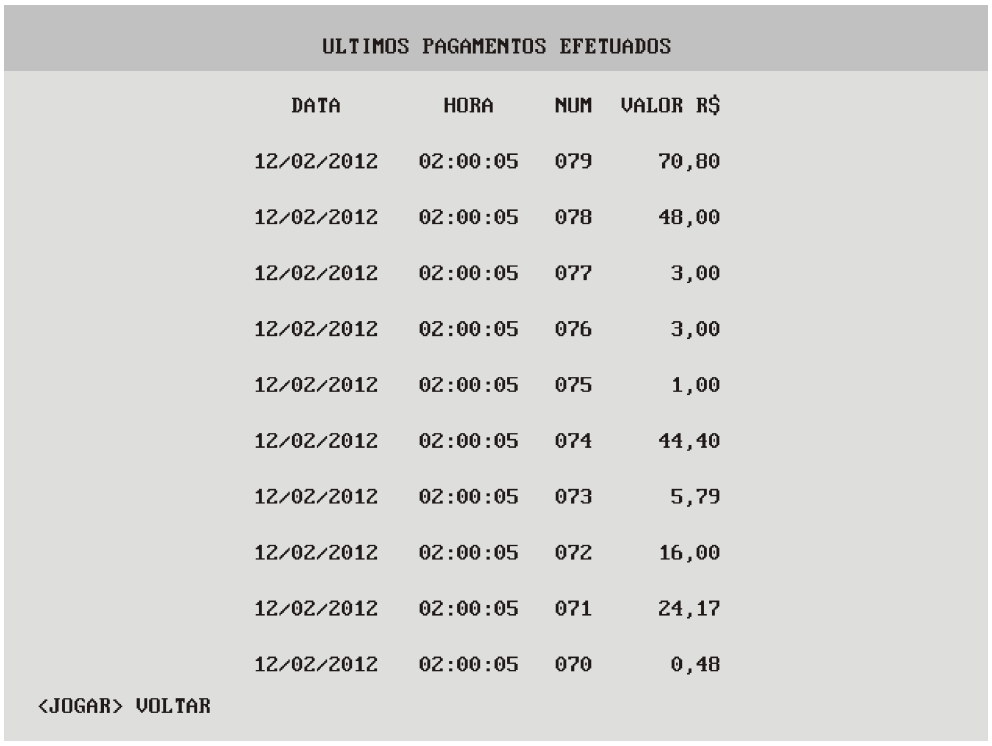

Figura 6.12. Tela dos últimos pagamentos efetuados.

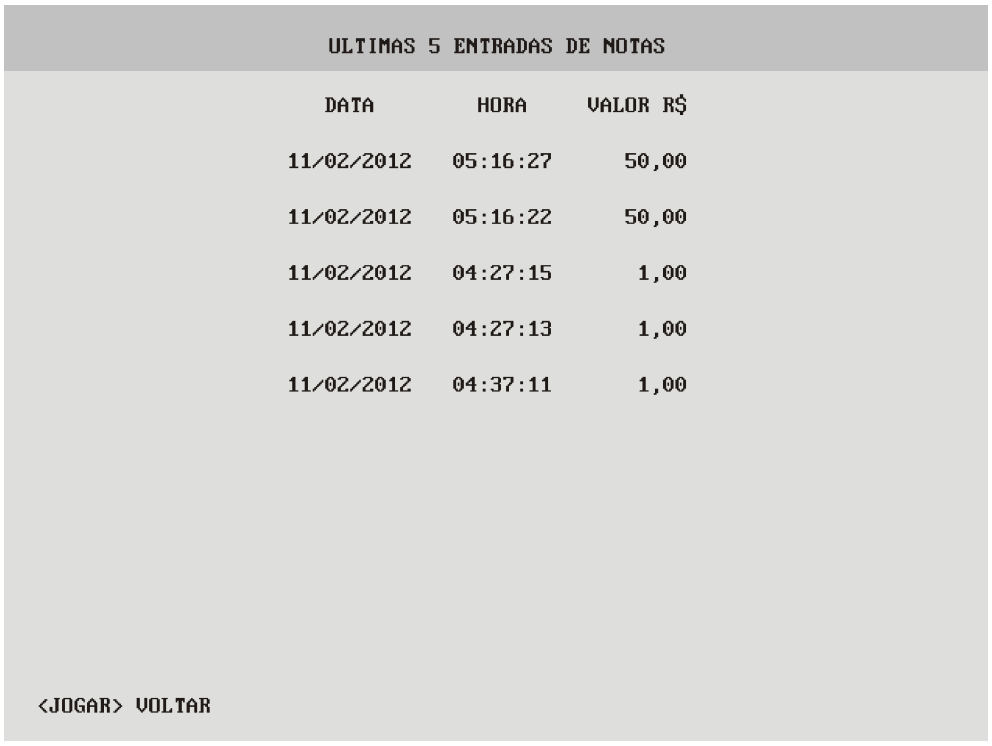

Figura 6.13. Tela das cinco últimas entradas de crédito.

A opção seguinte apresentada pela tela de configuração, acessível através da tecla “N”, refere-se à exibição individual dos dez últimos sorteios efetuados pelo equipamento, demonstrando graficamente a configuração final da tela principal quanto ao número de

linhas e modalidade de aposta escolhidos, bem como ao posicionamento das figuras em relação à matriz (Figura 6.3), possibilitando ao usuário avançar ou retroceder na sequência de exibição.

6.4.2.3. Segundo Nível de Segurança

Ao contrário das opções anteriores que realizam mera consulta aos valores armazenados na base principal de dados, ao informar o código correspondente ao pressionamento da tecla "T", a aplicação permite a alteração de alguns parâmetros de comportamento do equipamento. Porém, o acesso a essa funcionalidade é controlado por um mecanismo de checagem de senha a ser informada pelo usuário (Figura 6.14).

Figura 6.14. Tela de controle de acesso por senha de segurança.

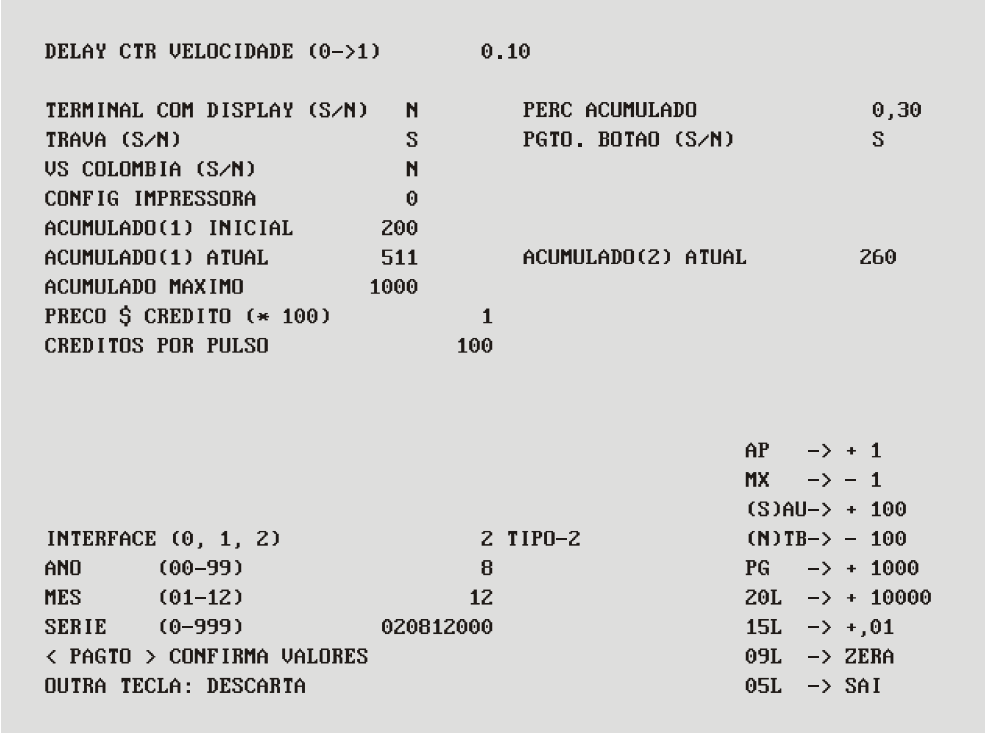

Figura 6.15. Tela de configuração de parâmetros acessível por senha de segurança.

Observando o comportamento da aplicação principal em relação ao valor de senha

informado pelo usuário, verificou-se que este é confrontado com uma cadeia de caracteres, com cinco dígitos de comprimento, armazenada também no próprio arquivo executável, porém divergente da senha utilizada para acessar a tela de leitura oficial.

Uma vez que a senha tenha sido informada corretamente, os parâmetros de configuração são exibidos um a um para que sejam ajustados conforme as instruções exibidas no quadrante inferior direito da tela (Figura 6.15). Todos os parâmetros são livremente configuráveis, exceto aquele intitulado “Config Impressora”.

6.4.2.4. Terceiro Nível de Segurança

Ao selecionar este parâmetro, ao invés de o valor informado ser armazenado no registrador correspondente, é realizada uma comparação com um terceiro valor armazenado no segmento de dados do arquivo, indicando tratar-se novamente de uma senha, divergente das duas senhas anteriores. Como exibido na Figura 6.15, ao pressionar a tecla “J”, o ajuste do parâmetro atual é descartado, mantendo o valor previamente armazenado.

6.4.2.5. Senhas de Acesso dos Níveis de Segurança

Diante da constatação da existência de três verificações distintas de senha (*hardcoded*) referentes aos respectivos níveis de segurança, ao realizar uma consulta ao resultado extraído pelas ferramentas Strings e IDA Pro de todos os arquivos “programa.exe” encontrados, verificou-se que a senha referente ao acesso à tela de configuração de parâmetros (Seção 6.4.2.3) é exibida sempre após a única ocorrência da cadeia de caracteres “informe a senha corretamente”.

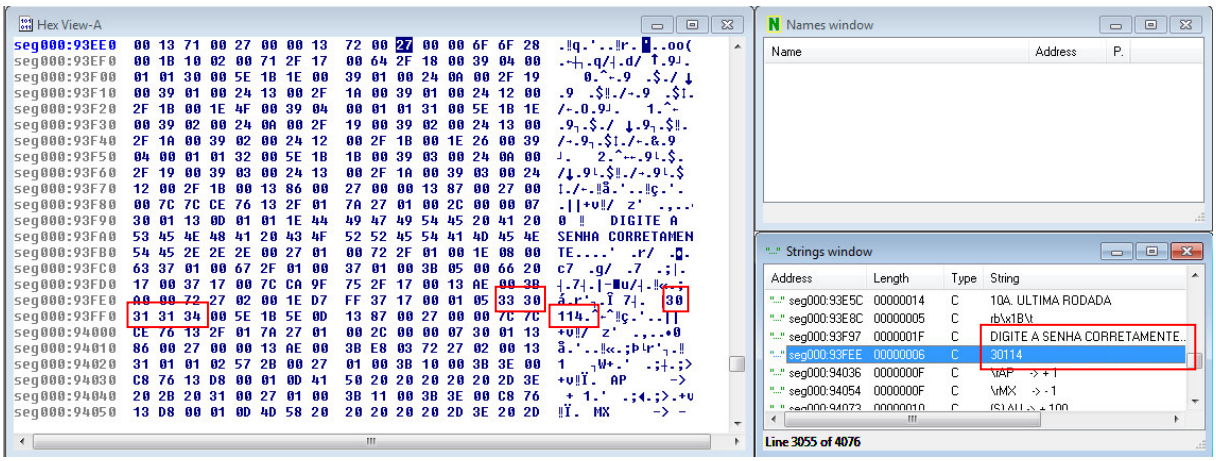

Figura 6.16. Localização da senha de acesso à tela de configuração de parâmetros.

A senha de acesso à tela de leitura oficial, por sua vez, é o primeiro valor numérico contendo cinco dígitos que antecede a cadeia de caracteres “Leitura Oficial” (Seção 6.4.2.2), assim como a senha de acesso ao item “Config Impressora” (Seção 6.4.2.5) é o primeiro valor numérico contendo cinco dígitos que sucede a cadeia de caracteres idêntica à descrição do item.

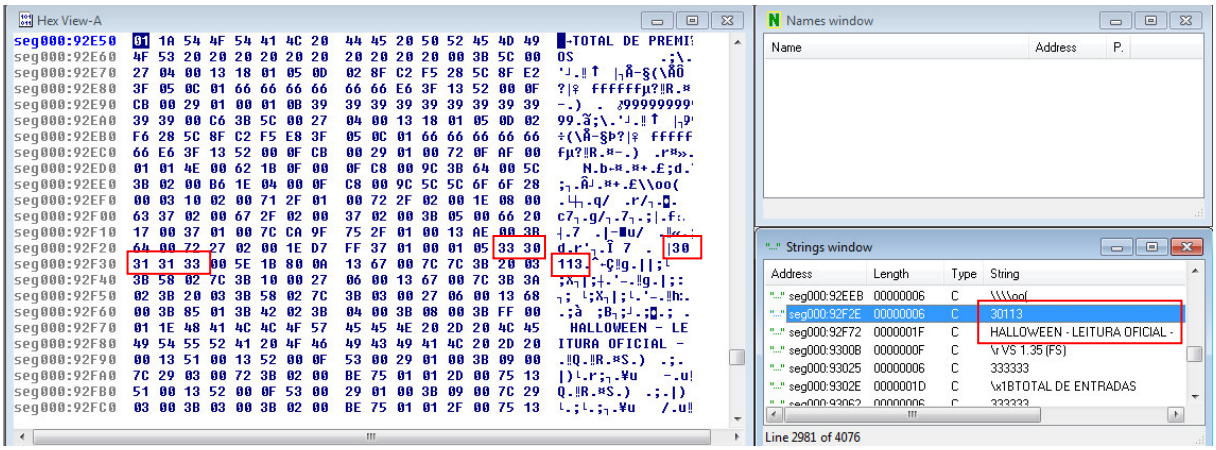

Figura 6.17. Localização da senha de acesso à tela de leitura oficial.

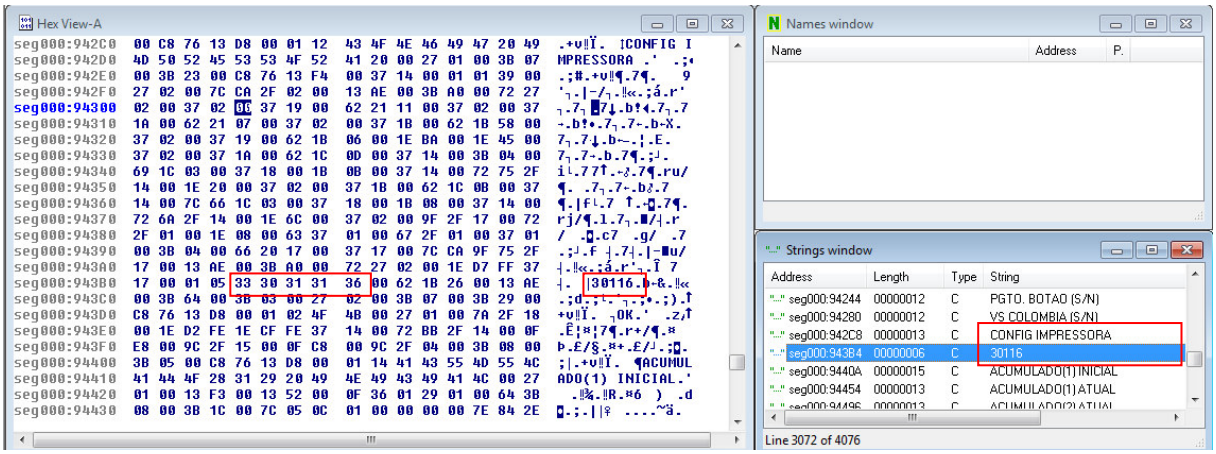

Figura 6.18. Localização da senha de acesso à tela “Config. Impressora”.

A localização das três senhas *hardcoded* na relação de cadeias de caracteres extraídas a partir do arquivo executável tende abreviar o tempo para obtenção dessa informação quando da análise de aplicativos semelhantes e que não sejam compatíveis com as senhas descritas no parágrafo anterior.

Porém, nos 23 (vinte e três) jogos foram identificadas as senhas conforme descrito na tabela a seguir, subdividindo os arquivos em quatro grupos distintos, de acordo com a convergência das três senhas.

Tabela 6.10. Relação de senhas encontradas nos 23 jogos examinados

Jogo	Senhas		
	Parâmetros	Leitura Oficial	Impressora
buca2.sys	30114	30113	30116
buca2ac.sys	30114	30113	30116
camp2ac.sys	30114	30113	30116
camp2007.sys	30114	30113	30116
fruta.sys	30114	30113	30116
fuga2acm.sys	30114	30113	30116
hallo2.sys	30114	30113	30116
hallo2ac.sys	30114	30113	30116
hallo2tk.sys	30114	30113	30116
sexy2.sys	30114	30113	30116
sexy2ac.sys	30114	30113	30116
vacalo2.sys	30114	30113	30116
hallofor.sys	30114	30118	30116
bigb1ac.sys	43210	01243	36546
fuga1ac.sys	43210	01243	36546
gcard1.sys	43210	01243	36546
goldcard.sys	43210	01243	36546
hallo1.sys	43210	01243	36546
hallo1ac.sys	43210	01243	36546
pantanal.sys	43210	01243	36546
trago1ac.sys	43210	01243	36546
vakalo1.sys	43210	01243	36546
oro2ac.sys	44220	01243	35666

6.4.3. Base de Dados Principal

Diante do fato de a base de dados principal, representada pelo arquivo “base.dbf” e que é armazenado no diretório “data”, apresentar conteúdo ininteligível, conforme mencionado anteriormente, procedeu-se com a modalidade de análise *online* com o intuito de verificar o comportamento da aplicação principal em relação aos valores armazenados.

6.4.3.1. Decodificação de Informações da Base

Durante a análise, constatou a existência de codificação das colunas originalmente definidas como do tipo “string” que armazenam valores numéricos. A codificação é trivial: cada byte que forma o número é registrado na forma da sua representação ASCII. Tendo em mente que a arquitetura x86 é *little endian*, a ordem é do menos significante para o mais significante. Como exemplo, considere o valor originalmente informado como “100”, que era gravado como “d” (a representação ASCII do número).

Assim, uma vez decodificados os valores armazenados nos equipamentos analisados, foi possível constatar a função das principais colunas da tabela abrigada pelo arquivo "base.dbf", conforme descrito na Tabela 6.10 a seguir.

Tabela 6.11. Dicionário de dados da tabela "base.dbf".

Coluna	Tipo	Permanente	Função
creatu	string[4]	N	Crédito inserido no equipamento
entc	string[4]	S	Total acumulado de entradas
entcpar	string[4]	S	Parcial acumulado de entradas
jack	string[4]	S	Total acumulado de pagamentos
jackpar	string[4]	S	Parcial acumulado de pagamentos
acapostas	string[4]	S	Total acumulado de apostas
acpremios	string[4]	S	Total acumulado de prêmios
acappar	string[4]	S	Parcial acumulado de apostas
acprpar	string[4]	S	Parcial acumulado de prêmios
creditom	string[4]	N	Crédito disponível no equipamento
bonusm	string[4]	N	Valor de bônus obtido na rodada
varacum	real	S	Valor base para cálculo do acumulado efetivo
aposta	char	N	Fator de aposta (1 a 10)
linhas	char	N	Quantia de linhas escolhidas
price	char	N	Preço de cada crédito na aposta atual (1-10)
rodada	string[4]	S	Contador de rodadas do equipamento
acumini	string[4]	S	Valor mínimo que deve ser acumulado
acumulado	real	S	Prêmio principal acumulado
acumbonus	real	S	Prêmio secundário acumulado
acumefect	real	S	Acumulado efetivo
acummax	string[4]	S	Valor máximo que pode ser acumulado
nserie	string[4]	S	Interface + ano + mês + série
pulsos	char	S	Quantidade de créditos para cada pulso
jogo01	string[24]	S	Armazena primeira última jogada
jogo02	string[24]	S	Armazena segunda última jogada
jogo03	string[24]	S	Armazena terceira última jogada
jogo04	string[24]	S	Armazena quarta última jogada
jogo05	string[24]	S	Armazena quinta última jogada
jogo06	string[24]	S	Armazena sexta última jogada
jogo07	string[24]	S	Armazena sétima última jogada
jogo08	string[24]	S	Armazena oitava última jogada
jogo09	string[24]	S	Armazena nona última jogada
jogo10	string[24]	S	Armazena décima última jogada
ctrdsp	Booleano	S	Display externo
ctrjack	Inteiro	S	Limita o acumulado ao valor de acummax
pago	string[90]	S	Armazena últimos 10 pagamentos da máquina
entcre	string[40]	S	Armazena últimos 5 créditos inseridos
seqjack	Char	S	Quantidade de pagamentos realizados
maca	Char	N	Valor de prêmio no bônus maça
caldeirao	Char	N	Valor de prêmio no bônus caldeirão

gameswon	string[4]	S	Quantidade de jogos ganhos no equipamento
colombia	Booleano	S	Desconsiderar relação crédito/moeda corrente
perce	Char	S	Percentual de retenção (impressora)
delay	Real	S	Controle de velocidade do jogo
fatoracum	Real	S	Percentual de incremento do acumulado
jog	Inteiro	S	(Não identificado)
crt	Char	N	(Não identificado)
trava	Booleano	S	Trava o bônus do caldeirão
pgtobtn	Booleano	S	Permite finalizar da sessão no equipamento

Ao examinar os itens elencados na Tabela 6.11, percebe-se que as colunas que trazem o termo "jack" referem-se sempre à ação de pagamento do "prêmio especial" ao apostador. "Jack" e "jackpar" armazenam os montantes referentes aos prêmios pagos pela máquina, sendo que o segundo é o valor parcial, pois pode ser zerado (Seção 6.4.2.2) a qualquer momento pelo administrador do sistema. "Seqjack", por sua vez, guarda a quantidade numérica de prêmios pagos desde a instalação do equipamento.

Com isso, ao examinar o comportamento da aplicação em relação ao valor armazenado em "ctrjack", verificou-se que este obriga que a máquina limite-se a acumular o valor definido em "acummax".

6.4.3.2. Parâmetros de Configuração Fraudulentos

As colunas "perce" e "trava" são aquelas que evidenciam o maior poder lesivo da aplicação principal. A primeira delas, definida na tela de configuração como "Config. Impressora", cujo acesso é realizado através de dupla verificação de senha, refere-se ao percentual de retenção do equipamento, aceitando valores numéricos entre 0 e 4. Quanto menor o valor informado, maior a retenção praticada pela máquina. Em todas as máquinas analisadas, o valor de "perce" estava configurado como "0".

Tal ajuste influencia negativamente na aleatoriedade do resultado, uma vez que a aplicação apresenta um laço de repetição que obriga a realização de novo sorteio caso o nível de retenção do equipamento não tenha sido atingido.

A coluna "trava", também configurada através da tela ilustrada pela Figura 6.15, quando marcada positivamente, impede a exibição das figuras compatíveis com as sequências de bônus ilustradas na Figura 6.4 referentes ao "caldeirão", sem prestar qualquer esclarecimento ao usuário da máquina, levando-o a erro.

Outro indício de fraude foi constatado na tela de bonificação denominada internamente

como “maçã” (Figura 6.5). Quando da montagem da interface em tempo de execução, os valores da matriz não são previamente sorteados. Ao contrário disso, o sorteio acontece tão somente quando um dos quadrantes é escolhido pelo apostador. Entretanto, assim como ocorre com o sorteio na tela principal, o resultado é confrontado com o nível de retenção do equipamento antes que este seja exibido ao usuário. Caso seja necessário, outro valor é sorteado sem que o usuário tenha ciência disso.

Através deste capítulo foi possível verificar que os programas de computador que gerenciam as máquinas caça-níquel analisadas utilizam mecanismos tanto para ofuscar informações referentes ao acesso a áreas restritas de leitura e configurações de parâmetros, como para dissimular alguns desses parâmetros de configuração, sugerindo um sentido diverso da sua funcionalidade original.

Assim, com base na adaptação do método utilizado por Vênere (2009) na análise de código malicioso para atender ao modelo proposto pelo NIST, infere-se a seguinte metodologia de análise de máquinas caça-níquel que pertençam ou não à família Halloween:

a) Devido à existência de parâmetros não identificados de configuração do ambiente operacional, que impediram a execução de uma imagem do dispositivo de armazenamento em um ambiente controlado, deve-se realizar uma cópia bit a bit do dispositivo de armazenamento, a fim de preservar a integridade do conteúdo armazenado no dispositivo original;

b) O equipamento deve ser iniciado com o dispositivo que contém a cópia bit a bit e deve-se realizar uma análise comportamental inicial com base na usabilidade da interface, onde serão identificadas as principais funcionalidades da aplicação;

c) De posse da análise comportamental, deve-se realizar um inventário dos itens de *hardware* do equipamento, identificando os componentes de interação com a aplicação, tais como dispositivos de entrada e saída. Durante a confecção do inventário, as funcionalidades identificadas na etapa anterior são mapeadas aos códigos produzidos pelos dispositivos de interação;

d) Tendo uma ideia formal inicial do comportamento da aplicação principal, deve-se realizar uma análise do ambiente operacional da aplicação, identificando arquivos responsáveis por subsidiar a execução do programa

que gerencia os dispositivos de interação. Nesta etapa são catalogados e separados os arquivos referentes à preparação do ambiente de suporte dos arquivos diretamente vinculados à aplicação;

e) Dentre os arquivos vinculados à aplicação, deve-se identificar aqueles que correspondem ao jogo propriamente dito e a possíveis base de dados utilizadas pela aplicação principal ou por aplicativos que deem suporte a ela, pois a existência de bases de dados sugere a utilização de parâmetros de configuração do ambiente e registro estatístico de jogadas;

f) Abrir as bases de dados encontradas e caso o conteúdo esteja codificado, tentar a decodificação delas através das fórmulas (6.1) e (6.2) propostas. Caso as fórmulas não sejam aplicáveis, buscar a decodificação pelo método de tentativa e erro ou pela análise *on-line* do código binário com o auxílio de um *debugger*;

g) Uma vez decodificadas as bases de dados, realizar um mapeamento inicial entre os valores armazenados, referentes a parâmetros de configuração do ambiente e registro estatístico de jogadas, e os elementos apresentados pela interface;

h) Caso os arquivos referentes a jogos fornecidos pela máquina estejam na sua forma binária, abri-lo através de uma ferramenta *debugger* e realizar a análise *on-line* retificando ou ratificando o mapeamento inicial realizado entre os valores armazenados na base de dados correspondente e os elementos de entrada e saída apresentados pela interface da aplicação principal;

i) Caso os jogos tenham sido comprimidos com a utilização de senha, verificar se algumas das bases de dados identificadas possui as senhas de descompressão armazenadas e realizar a descompactação. Por sua vez, caso os jogos tenham sido submetidos a uma ação de empacotamento, utilizar uma ferramenta de *unpacking* para ter acesso ao código-binário original ao invés do código-binário do desempacotador. Em alguns casos, o arquivo principal pode ter sido submetido primeiramente a uma função de empacotamento e posteriormente a uma função de compressão. Neste caso, deve-se realizar primeiramente a descompressão e em seguida o

desempacotamento;

j) Durante a análise *on-line* deve-se procurar por funções de captura de texto e geração de números randômicos em arquiteturas de 32-bits ou superiores, ou interrupções de relógio interno e teclado em arquiteturas de 16-bits, para facilitar a identificação de áreas ocultas de configuração de parâmetros ou das evidências da prática de jogo de azar, respectivamente;

k) Ainda durante a análise *on-line*, quando da identificação de regiões obscuras de acesso a configurações de parâmetros com verificação de senha, buscar pela existência de senhas *hardcoded* gravadas na área de dados do código binário ou simplesmente realizar a alteração do *flag* responsável por decidir o desvio adotado por determinado salto condicional presente no código diante da verificação da senha de acesso;

l) Realizar testes de operação, ajustando os parâmetros e observando a mudança de comportamento da aplicação, a fim de produzir um modelo comportamental mais apurado. Esse modelo visa constatar as fraudes provenientes da dissimulação de funcionalidades identificadas na etapa de análise inicial do comportamento e que induzem o apostador a acreditar que o equipamento comportar-se-á de maneira distinta daquela que foi programada.

7. SIMILARIDADE ENTRE JOGOS

A metodologia proposta no Capítulo 6, apesar de possibilitar a identificação de um modelo comportamental da aplicação que gerencia as máquinas caça-níqueis, é um tanto quanto dispendioso de ser aplicado a todas as máquinas apreendidas em uma operação policial.

Assim, este capítulo propõe técnicas de comparação de executáveis que vão da mais simples a mais complexa, abreviando o tempo de análise de jogos aparentemente distintos, mas que possuem executáveis internamente idênticos, semelhantes em relação às instruções binárias ou completamente divergentes quanto à localização das instruções binárias, mas semelhantes quanto aos blocos de função e suas dependências internas.

Embora todos os vinte e três jogos examinados apresentem usabilidade idêntica - salvos pequenos ajustes de versão supostamente atribuídos a correções de *bugs*[13], compartilhem do mesmo ambiente operacional e adotem comportamento interno semelhante em relação às colunas da base de dados principal, observou-se tanto uma divergência quanto ao tamanho do arquivo executável, como também quanto à quantidade total de imagens utilizadas para compor a interface.

A divergência entre a quantidade de arquivos utilizados pela interface de cada jogo influencia diretamente na discrepância constatada em relação ao tamanho dos arquivos compactados, apresentados com a extensão “sys”, levando a uma falsa impressão de que se tratam de aplicações completamente incompatíveis.

Porém, o que se observa é que tanto o arquivo executável de um jogo pode ser substituído pelo arquivo de quaisquer um dos outros jogos, como também a característica da interface ser construída dinamicamente com base em um conjunto de imagens, permite a criação de um novo jogo. Para isso, basta que sejam substituídos os arquivos originais por outros com mesmo nome, dimensão e profundidade de *bits*, porém que ilustre a temática desejada.

[13] Falhas identificadas em programas de computador oriundas do seu desenvolvimento.

Tabela 7.1. Comparativo entre os 23 jogos examinados.

Padrão	Arquivo original	Executáveis	Tamanho	Arquivos
1	bigb1ac.sys	programa.exe	273 KB	339
	fuga1ac.sys	programa.exe	273 KB	372
	gcard1.sys	programa.exe	273 KB	333
	goldcard.sys	programa.exe	273 KB	333
	hallo1.sys	programa.exe	273 KB	335
	hallo1ac.sys	programa.exe	273 KB	335
	pantanal.sys	programa.exe	273 KB	373
	trago1ac.sys	programa.exe	273 KB	331
	vakalo1.sys	programa.exe	273 KB	331
2	buca2.sys	programa.exe	281 KB	335
	buca2ac.sys	programa.exe	281 KB	341
	camp2ac.sys	programa.exe	281 KB	314
	camp2007.sys	programa.exe	281 KB	321
	fruta.sys	programa.exe	281 KB	315
	fuga2acm.sys	programa.exe	281 KB	376
	hallo2.sys	programa.exe	281 KB	335
	hallo2ac.sys	programa.exe	281 KB	335
	hallo2tk.sys	programa.exe	281 KB	331
	sexy2.sys	programa.exe	281 KB	333
	sexy2ac.sys	programa.exe	281 KB	373
	vacalo2.sys	programa.exe	281 KB	374
3	hallofor.sys	programa.exe	677 KB	335
	oro2ac.sys	programa.exe	677 KB	236

Ao serem classificados com base no tamanho apresentado pelo executável (Tabela 7.1), é possível constatar três grupos distintos. Aqueles com tamanho de 273 KB correspondem a uma versão do aplicativo dotado de apenas um valor acumulado (Figura 7.1) - embora a base de dados de todas as máquinas apresentem duas colunas para armazenamento de acumulados - conforme constatado pelo acréscimo do sufixo “1ac” ao nome da maioria dos jogos.

Figura 7.1. Arquivo “tela1.dat” presente no arquivo compactado “bigb1ac.sys”

Por sua vez, os arquivos com tamanho de 281 KB, geralmente dotados do sufixo “2ac”, oferecem dois valores acumulados (Figura 7.2) para pagamento de premiação.

Figura 7.2. Arquivo “tela1.dat” presente no arquivo compactado “buca2.sys”

Apesar da divergência de tamanho existente entre os arquivos do padrão 3 e os demais arquivos, observa-se que o arquivo “tela1.dat”, que integra o jogo cujo arquivo compactado denomina-se “hallofor.sys” (Figura 7.3) e o arquivo “tela.dat”, que corresponde à interface principal do jogo distribuído através do pacote “oro2ac.sys” (Figura 7.4), apresentam um e dois acumulados respectivamente. Tal semelhança sugere que eles correspondem ao padrão descrito anteriormente que melhor se enquadra na aparência da sua interface.

A divergência existente entre o nome do arquivo referente à imagem de pano de fundo do jogo distribuído com o arquivo “oro2ac.sys” (tela.dat) e os demais jogos (tela1.dat) justifica-se pelo posicionamento das informações referentes à “crédito”, “prêmio”, “aposta”, “valor pago” e “acumulados” na interface (Figura 7.4). O fato de os arquivos binários poderem ser trocados entre si indica que a aplicação faz verificação da existência do arquivo “tela.dat”, dando preferência a ele,

Em caso positivo, carrega-o como pano de fundo e concentra a exibição das informações no terço inferior da tela (Figura 7.4). Caso contrário, verifica a existência do arquivo “tela1.dat”. Se disponível, utiliza-o como pano de fundo e divide a exibição das informações entre os terços superior e inferior da interface (Figura 7.3).

Figura 7.3. Arquivo “tela1.dat” presente no arquivo compactado “hallofor.sys”

Figura 7.4. Arquivo “tela.dat” presente no arquivo compactado “oro2ac.sys”

7.1. COMPARAÇÃO POR VALORES DE HASH

A primeira abordagem empregada para identificar similaridade entre arquivos foi a utilização de função de hash diretamente sobre os arquivos binários. Uma função de hash consiste de um cálculo matemático que, a partir de uma entrada arbitrária de dados, produz um resultado de tamanho fixo (Schneier, 2003, p.83). Como o resultado é obtido através de cálculos matemáticos, duas entradas idênticas devem resultar no mesmo valor de hash. Em contrapartida, a alteração de um único byte do arquivo binário implica que um valor de hash completamente diferente seja calculado.

Dentre as propriedades das funções de hash definidas por Schneier (2003, p.84-5), a única aplicável ao propósito deste trabalho é a resistência à colisão. Devido ao fato de que há infinitas entradas de dados, nenhuma função de hash será totalmente livre de colisão, já que a quantidade de combinações possíveis de saída é finita. Com isso, apesar da baixa probabilidade de ocorrência, o método de identificação de similaridades por funções de hash pode fornecer tanto resultado falso-positivos (colisão).

Mesmo com essa propriedade, as funções de hash são amplamente utilizadas em

ferramentas forenses, como o FTK[14] (*Forensic Toolkit*), por exemplo.

Para geração dos valores de hash observados na Tabela 7.2 empregou-se a função MD5 (*Message Digest 5*). Para permitir a fácil identificação dos arquivos executáveis, já que todos apresentam o mesmo nome, adotou-se o nome do arquivo que representa o pacote original do jogo, acrescido do prefixo "p".

Tabela 7.2. Valores de hash gerados com base nos arquivos binários

Padrão	Arquivos	Qtd	MD5	Tamanho
1	pbigb1ac.exe pfuga1ac.exe pgcard1.exe pgoldcard.exe phallo1.exe phallo1ac.exe ppantanal.exe ptrago1ac.exe pvakalo1.exe	9	6d9bd836bb4ca51be0ade867db6f458f	279.472
2	buca2ac.sys fuga2acm.sys hallo2.sys hallo2ac.sys hallo2tk.sys vacalo2.sys	6	5731b6ea027f70d4c51d5cb6e13b25a4	287.712
3	psexy2.sys pSexy2ac	2	20368ff05697fcf81f95197d4f63cbdd	287.712
4	pBuca2	1	9df32ecd8b2686e9211cd0adbd622c9f	287.712
5	pCamp2007	1	9b64b7faf46069db203c22dcc9ea3734	287.712
6	pCamp2ac	1	106f21cbcc9c3b3bd1483623d1ce3ab7	287.712
7	pFruta	1	2a4f49a2f034524ffac4c94ba0372ce4	287.712
8	pHallofor	1	e75267562758fe6f06efe7b63a0f1ad8	692.559
9	pOro2ac	1	76fa43689fbdc24ca8e3840480100d8a	692.543

Com base nos valores de hash apresentados na Tabela 7.2, constata-se que todos os jogos cujo executável possuía tamanho de 273 KB apresentaram o mesmo valor de hash. A comparação baseada nos valores de hash resultou em 09 (nove) grupos de arquivos classificados como "não-idênticos".

Essa convergência de valor indica que todos os jogos possuem uma versão idêntica do arquivo "programa.exe". A aparente divergência apresentada pelos jogos está unicamente no conjunto de imagens (com temática distinta) utilizado na composição da interface em

[14] www.accessdata.com

tempo de execução.

7.2. COMPARAÇÃO BLOCO A BLOCO DE EXECUTÁVEIS

Diante da não adequação das funções de hash para identificação de arquivos binários semelhantes, já que mesmo pequenas alterações resultam em valores de hash completamente diferentes, foi adotada a abordagem de comparação bloco a bloco. Para tanto, utilizou-se a ferramenta nativa FC (*File Compare*) do sistema operacional Windows (Microsoft, 2012).

Na comparação bloco a bloco, cada byte do arquivo original é comparado com a posição correspondente no arquivo questionado. Em caso de divergência, é adicionado ao resultado o endereço da posição e os valores apresentados pelos bytes de ambos os programas. O resultado final é a lista com a quantidade total de divergências encontradas.

Tabela 7.3. Relacionamento das diferenças em byte de cada grupo da Tabela 7.2

Padrão	1	2	3	4	5	6	7	8	9
1		267.717	267.715	267.717	267.719	267.717	267.718	276.869	276.871
2	267.717	0	90	91	63	91	45	284.985	284.985
3	267.715	90	0	33	119	32	117	284.984	284.984
4	267.717	91	33	0	118	27	120	284.985	284.985
5	267.719	63	119	118	0	118	27	284.986	284.986
6	267.717	91	32	27	118	0	120	284.985	284.985
7	267.718	45	117	120	27	120	0	284.986	284.986
8	276.869	284.985	284.984	284.985	284.986	284.985	284.986	0	310.362
9	276.871	284.985	284.984	284.985	284.986	284.985	284.986	310.362	0

Ao examinar a Tabela 7.3 é constatado um pequeno percentual de discordância entre os arquivos referentes aos padrões 2 a 7. O maior valor de discordância encontrado (120), existente entre os padrões 4 e 7 e 6 e 7, representa pouco mais que 0,04% do total de bytes do arquivo.

Apesar de não apresentarem o mesmo valor de hash, é possível garantir, com alto grau de certeza, que os arquivos binários referentes àqueles padrões correspondem a uma versão com funcionalidades altamente semelhantes. Essa pequena discrepância pode ser atribuída à personalização do título das telas de administração do sistema (Figura 6.9), tela de leitura parcial (Figura 6.10) e tela de leitura oficial (Figura 6.11), dentre outras informações que não afetam o comportamento da aplicação.

Da comparação bloco a bloco, o conjunto de padrões foi reduzido a 04 (quatro).

7.3. COMPARAÇÃO ISOMÓRFICA DE GRAFOS

Diante da grande divergência de tamanho apresentada, que sugere que os jogos referentes ao padrão 3 da Tabela 7.2 possuem quantidade superior ao dobro do código binário dos jogos dos demais padrões, foi realizada a comparação entre a representação em grafos dos quatro jogos aparentemente distintos (Flake, 2004 e Dullien & Rolles, 2005).

Tal representação em grafo, construída com base no relacionamento existente entre as funções presentes na aplicação, foi obtida através da ferramenta IDA Pro. Os arquivos foram abertos como "executável do MS-DOS" e foi optado pela exibição das informações extras existentes depois da marcação do término de código.

Aberto o arquivo "programa.exe", foi necessário posicionar o cursor no início do segundo segmento de código e optar pela exibição das informações na modalidade "código" (opção edit/code do menu), ao invés da exibição delas na modalidade "dados". Esta última modalidade foi a configuração-padrão adotada pela ferramenta quando da abertura dos arquivos. A configuração em questão é necessária para que a ferramenta IDA Pro realize a análise automática das chamadas de função presentes no código e confeccione a exibição gráfica do relacionamento existente entre elas.

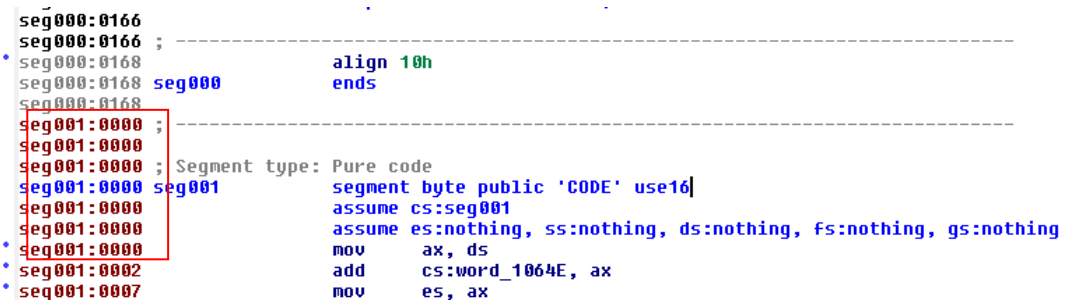

Figura 7.5. Identificação do segmento de código na ferramenta IDA Pro

Quando da abertura dos arquivos pertencentes aos padrões 1 e 2 da Tabela 7.2, verificou-se que os códigos binários daqueles jogos apresentam 19 (dezenove) funções, sendo uma a função principal, identificada como "start" e as demais sub-rotinas identificadas pelas instruções *assembly* "proc" e "end p".

Porém, os arquivos pertencentes ao padrão 3, descrito na mesma tabela, apresentam uma lista de aproximadamente 1.700 funções, indicando que, com exceção dos jogos

representados pelos arquivos “hallofor.sys” e “oro2ac.sys”, os arquivos executáveis dos demais jogos foram submetidos a algum algoritmo de empacotamento de executáveis.

O emprego de técnicas de empacotamento, além de reduzir o tamanho do arquivo binário (Fergotech, 2012), tem a propriedade de ofuscar o código original. Durante o empacotamento, seções são reorganizadas devido à exclusão de seções vazias. O empacotamento de funções que possuem tal característica e a inserção de rotinas de desempacotamento, também influencia na transformação da sua estrutura (Hardware.com.br, 2012). Logo, o grafo exibido para os arquivos compactados (Figura 7.2) não refletirá a estrutura original do arquivo.

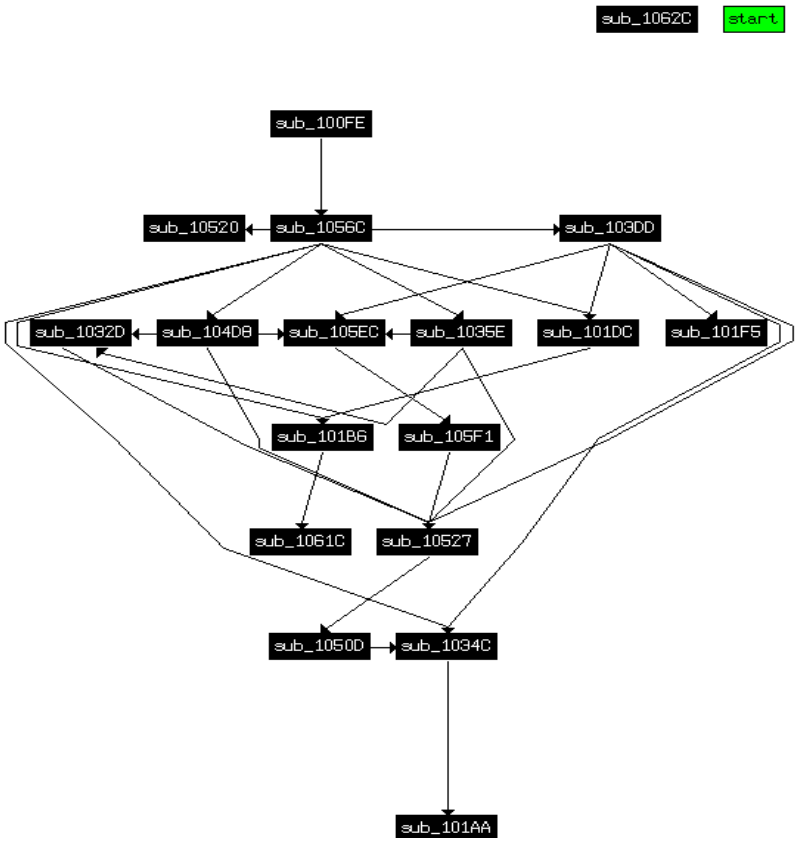

Figura 7.6. Grafo das funções encontradas no arquivo binário do pacote “bigb1ac.sys”.

Assim, os arquivos que apresentaram grafo semelhante ao ilustrado na Figura 7.6, antes de terem sua estrutura interna confrontada com a dos demais jogos, foram submetidos à ação da ferramenta de descompressão Cup386[15], que oferece suporte à maioria dos formatos conhecidos. Com isso, os arquivos pertencentes aos padrões 1 e 2, que apresentavam uma lista reduzida de funções, passaram a ter uma quantidade de funções compatível com os

[15] Descompactador de arquivos binários desenvolvido pela Cyberware

arquivos pertencentes ao padrão 3 (Tabela 7.4).

Para permitir a fácil identificação dos arquivos executáveis, já que todos apresentam o mesmo nome, adotou-se o nome do arquivo que representa o pacote original do jogo, adicionando o prefixo “u” ao arquivo gerado no desempacotamento.

Tabela 7.4. Quantidade de funções identificadas nos arquivos desempacotados

Padrão	Arquivo	Original	Tamanho	Funções
1	ubigb1ac.exe	programa.exe	667 KB	1644
	ufuga1ac.exe	programa.exe	667 KB	1644
	ugcard1.exe	programa.exe	667 KB	1644
	ugoldcard.exe	programa.exe	667 KB	1644
	uhallo1.exe	programa.exe	667 KB	1644
	uhallo1ac.exe	programa.exe	667 KB	1644
	upantanal.exe	programa.exe	667 KB	1644
	utrago1ac.exe	programa.exe	667 KB	1644
	uvakalo1.exe	programa.exe	667 KB	1644
2	ubuca2.exe	programa.exe	675 KB	1646
	ubuca2ac.exe	programa.exe	675 KB	1646
	ucamp2ac.exe	programa.exe	675 KB	1646
	ucamp2007.exe	programa.exe	675 KB	1646
	ufruta.exe	programa.exe	675 KB	1646
	ufuga2acm.exe	programa.exe	675 KB	1646
	uhallo2.exe	programa.exe	675 KB	1646
	uhallo2ac.exe	programa.exe	675 KB	1646
	uhallo2tk.exe	programa.exe	675 KB	1646
	usexy2.exe	programa.exe	675 KB	1646
	usexy2ac.exe	programa.exe	675 KB	1646
	uvacalo2.exe	programa.exe	675 KB	1646
3	phallofor.exe	programa.exe	677 KB	1649
4	poro2ac.exe	programa.exe	677 KB	1647

Além da quantidade de funções e do tamanho físico compatível, os arquivos descompactados apresentaram grafo com forma também semelhante (Figura 7.3 e Figura 7.4). Porém, devido à grande quantidade de funções identificadas, a visualização do grafo dirigido ficou prejudicada.

Figura 7.7. Grafo dirigido do arquivo binário do pacote “bigb1ac.sys” depois de

descompactado.

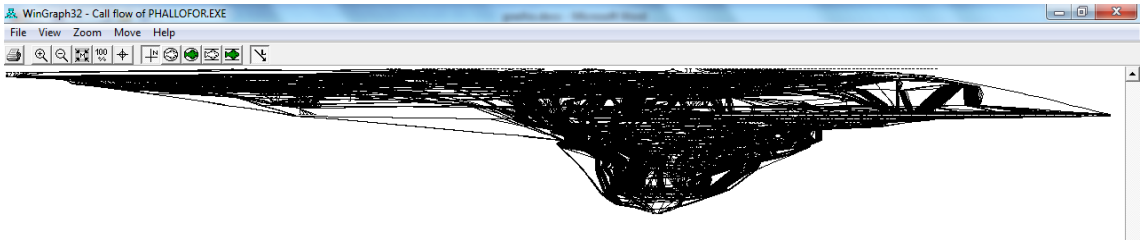

Figura 7.8. Grafo dirigido das funções do arquivo binário do pacote “hallofor.sys”.

Foi procedida então com a comparação dos grafos de controle de fluxo (*cfg*) de algumas das funções identificadas no código (Figura 7.9 e 7.10). A escolha das funções deu-se pela grande quantidade de argumentos declarados, por entender que tal característica implica em uma maior funcionalidade. Embora fosse impraticável a análise manual da totalidade das funções apresentadas pela aplicação, a convergência daquelas de maior volume com a forma gráfica apresentada pelos grafos dirigidos (Figura 7.7 e 7.8), permite inferir que ambos os arquivos foram produzidos a partir de um mesmo projeto de origem.

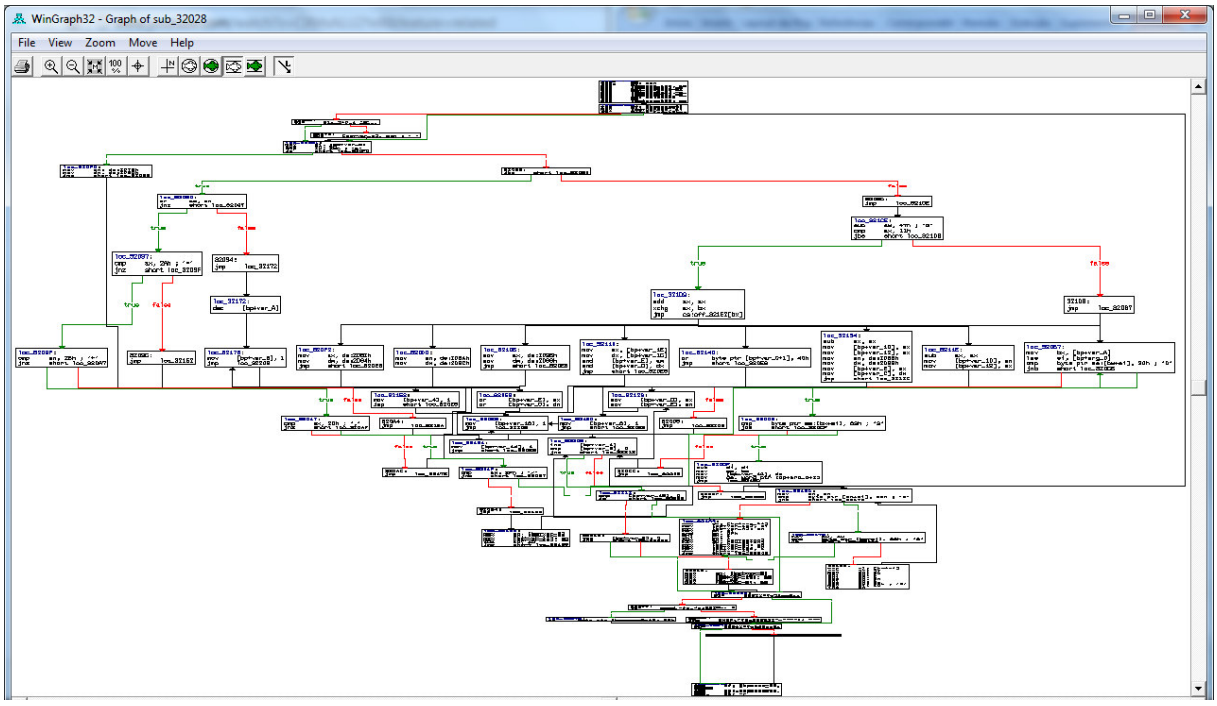

Figura 7.9. Grafo de controle de fluxo da função “sub_32028” do arquivo binário referente ao grafo dirigido da Figura 7.7.

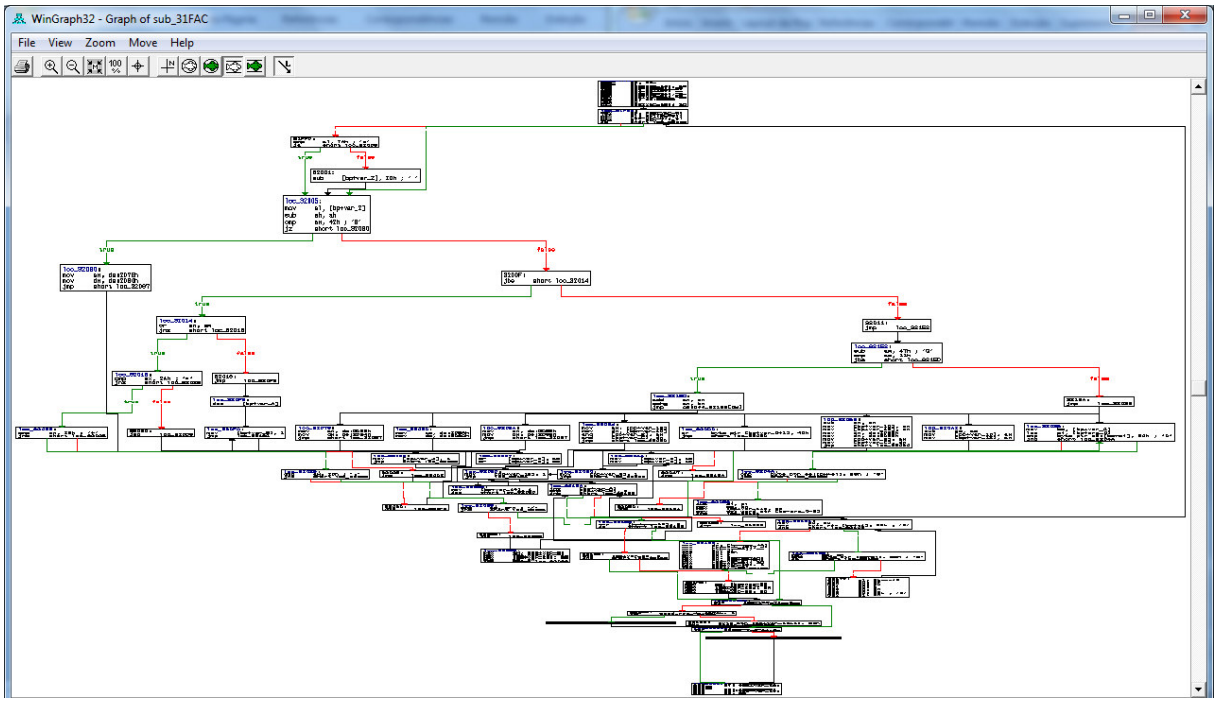

Figura 7.10. Grafo de controle de fluxo da função “sub_31FAC” do arquivo binário referente ao grafo dirigido da Figura 7.8.

A análise dos grafos de controle de fluxo das funções de arquivos apresentados como distintos através das abordagens “verificação de hash” e “comparação bloco a bloco”, mostrou que trechos relevantes de ambos os arquivos binários apresentam estrutura idêntica.

A pequena divergência na quantidade de funções entre os arquivos analisados é atribuída a implementação de funcionalidades de categorias distintas como o pagamento de um ou dois prêmios especiais (bônus). Alterações também podem ocorrer diante da correção de falhas de desenvolvimento (*bugs*), uma vez que são amplamente comercializadas, de forma clandestina, listas contendo eventuais falhas das primeiras versões desses programas.

A divergência de nomes entre as funções, a despeito da sua estrutura interna equivalente, evidencia que houve uma reorganização do código, pois os nomes são atribuídos em alusão ao endereçamento. Tal divergência implicou nos resultados falso-negativos apresentados pelos métodos anteriores quanto à similaridade. Porém, durante a análise, nenhum dos dois métodos apresentou resultado falso-positivo.

Diante disso, constata-se que a comparação isomórfica de grafos de Flake (2004) e Dullien & Rolles (2005) complementa as duas abordagens anteriores, pois é capaz de solucionar resultados falso-negativos apresentados por elas.

Ao final deste capítulo é possível verificar que a melhor abordagem de comparação de arquivos executáveis é a união de todas as técnicas apresentadas, partindo da mais simples para a mais complexa. Para tanto, deve-se realizar primeiramente a verificação dos valores de *hash*, já que é a técnica mais rápida de identificar arquivos cujo código binário seja idêntico.

Em seguida, por se tratar de técnica de baixo grau de complexidade e que não necessita de ferramenta de terceiros, deve-se executar a comparação bloco a bloco de executáveis, pois serão identificados somente os blocos divergentes entre os arquivos. Um pequeno percentual de blocos divergentes em relação ao tamanho do arquivo indica conteúdo interno com alto grau de semelhança, apenas com pequenas alterações não significativas, como correções de pequenos *bugs*.

Somente depois de descartadas as duas técnicas anteriores deve-se realizar a análise do isomorfismo do grafo dirigido e do grafo de controle de fluxo das funções de ambos os executáveis.

8. CONCLUSÕES

Diante da incerteza que paira sobre o comportamento interno das máquinas caça-níquel simuladas por programas de computador, em todo o território brasileiro tem havido divergência no entendimento dos magistrados quanto ao seu enquadramento ou não no rol dos jogos de azar. Por simularem certa jogabilidade, desperta a falsa impressão de que a habilidade do apostar pode influenciar na probabilidade de ganho ou perda.

Com o objetivo de pacificar a discussão, o presente trabalho aplicou técnicas de Engenharia Reversa adaptadas da metodologia proposta por Vênere (2009, p.118-143) para alcançar as duas primeiras etapas do modelo de Reengenharia de Software proposto pelo NIST (Peters & Pedrycz, 2001).

A partir delas foi possível realizar a análise do comportamento geral da aplicação das máquinas tipo "Halloween" desde o momento em que o apostador aciona o botão que dispara o mecanismo de sorteio, até o momento em que o resultado é exibido na tela. O estudo foi aplicado sobre um conjunto de 08 (oito) máquinas apreendidas pela Seção de Criminalística de Ji-Paraná, que apresentaram as características desejadas para a pesquisa. Nos dispositivos de armazenamento das máquinas foram identificadas 23 (vinte e três) versões aparentemente distintas da aplicação principal.

Durante a análise foram encontrados parâmetros de configuração ocultos ao apostador com até três níveis de segurança por verificação de senha de acesso (*hardcoded*). Ao relacionar esses parâmetros com o comportamento da aplicação, percebeu-se que alguns deles influenciam diretamente no percentual de retenção (lucro) da máquina e no pagamento de premiação. Foram evidenciadas ainda dissimulações de arquivos e de opções de configuração, como por exemplo, o parâmetro apresentado como "configuração da impressora", que na verdade corresponde ao fator de retenção.

Embora o sorteio seja realizado de forma aleatória, sem qualquer hipótese de o apostador influenciar no resultado depois de disparada a ação correspondente, foram constatados parâmetros de configuração (Tabela 6.11) que impedem que uma determinada sequência de premiação seja exibida na tela. Com isso, é evidenciado não só a prática de jogos de azar (aleatoriedade do sorteio) como também a possibilidade de fraude (bloqueio não transparente de opção de premiação).

Uma vez consolidada a análise do comportamento da aplicação, foram adotadas três

abordagens distintas para identificação de similaridade entre as diferentes versões disponíveis. Na verificação de similaridade por valor de *hash*, o conjunto inicial de 23 (vinte e três) versões, foi reduzido a 09 (nove). Em seguida, adotando a abordagem de comparação bloco a bloco, foram identificados resultados falso-negativos referentes a abordagem anterior. Com isso, o conjunto foi novamente reduzido. Desta vez, para 04 (quatro) versões aparentemente distintas.

Ao submeter o conjunto restante à abordagem de comparação isomórfica de grafos de funções (Flake, 2004 e Dullien & Rolles, 2005), verificou-se inicialmente que alguns dos arquivos binários foram submetidos a um algoritmo de empacotamento e por este motivo apresentavam discrepância quanto ao tamanho físico. Depois de desempacotados, os quatro padrões restantes apresentaram-se com alto grau de semelhança, evidenciada pela forma gráfica dos respectivos grafos dirigidos e dos grafos de controle de fluxo (*cfg*) das funções com maior quantidade de blocos de controle.

As contribuições do presente trabalho têm seu foco na caracterização da prática de jogo de azar ou fraudes. São elas:

a) Caracterização do comportamento interno dos programas de computador que controlam as máquinas tipo “Halloween”, para subsidiar a instrução de laudos periciais confeccionados por Peritos Criminais ao analisar jogos idênticos aos abordados pelo presente trabalho;

b) Apresentação de abordagens de identificação de similaridades entre aplicativos aparentemente distintos que visam auxiliar nos exames de versões de jogos das máquinas tipo “Halloween” não abrangidos pelo presente trabalho;

c) Apresentação de uma metodologia de análise que permite aos Peritos Criminais realizar, por analogia, o exame de programas de computador que controlem máquinas incompatíveis com o tipo “Halloween”.

8.1. DIFICULDADES ENCONTRADAS

Embora seja contemporâneo, o jogo que controla as máquinas caça-níquel que integram o objeto estudo do presente trabalho foi desenvolvido em uma arquitetura obsoleta, cujas características não mais são suportadas pela maioria das ferramentas atuais destinadas à tarefa de engenharia reversa, que por si só possui um grau elevado de complexidade.

Aliado ao sistema de proteção contra cópia não autorizada desenvolvido pelo fabricante da aplicação, as tentativas de execução das imagens RAW, extraídas dos dispositivos de armazenamento, em um ambiente controlado oferecido por uma máquina virtual, não apresentaram resultado satisfatório devido a alguma incompatibilidade não identificada existente entre o *hardware* emulado e os requisitos apresentados pela aplicação.

Diante disso, foi necessário copiar o debbugger CodeView para um diretório criado no dispositivo de armazenamento, acionar o *prompt* de comandos a partir da tela inicial de escolha de jogos (Figura 6.4), executar novamente o arquivo "menu.exe" para descompactar o jogo desejado e só então abri-lo através da ferramenta.

8.2. TRABALHOS FUTUROS

Face à complexidade apresentada pela análise manual das funções, sugere-se como trabalho futuro a construção de uma ferramenta que agregue as três abordagens de identificação de similaridades em aplicações aparentemente distintas, mantendo um registro de *hash* das versões analisadas para instruir exames futuros.

Outra possibilidade de trabalho futuro é o aprofundamento da pesquisa referente ao comportamento interno da aplicação para instruir o desenvolvimento de uma ferramenta que realize a aferição automática dos percentuais de ganho ou perda do equipamento. De forma completamente automatizada, a aplicação realiza *n* interações com o jogo e verifica os resultados gerados, mantendo um histórico independente da relação aposta x ganho, otimizando a abordagem proposta por Nogueira (2002) em seu estudo.

REFERÊNCIAS BIBLIOGRÁFICAS

Allslotgames. (2012). How the slot machine works? Disponível em: www.allslotsgames. com/how_slots_work.php. Acesso em: 7 de janeiro de 2012.

Amwatech. (2011a). Emulador de Teclado AVD005. Disponível em: www.amwatech.com. br/pdfs/manual_AVD005.pdf. Acesso em: 17 de agosto de 2011.

Amwatech. (2011b). Emulador de Teclado HW002. Disponível em: www.amwatech.com. br/pdfs/manual_ hw002.pdf. Acesso em: 17 de agosto de 2011.

Aurélio. (2011). Informática. Disponível em: www.dicionariodoaurelio.com. Acesso em: 18 de agosto de 2011.

Bello, Leo. (2008). Aprendendo a Jogar Poker: Princípios, Técnicas e Prática (2.ed). Rio de Janeiro: Nova Fronteira.

Casey, Eoghan., Malin, Cameron H. & Aquilina, James M. (2008). Malware Forensics: Investigating and Analyzing Malicious Code. Syngress: Waltham MA.

Clube do Hardware. (2001). Arquitetura de 64 bits da AMD. Disponível em: www.clubedo hardware.com.br/artigos/Arquitetura-de-64-bits-da-AMD-x86-64/376/3. Acesso em: 05 de janeiro de 2012.

Conjur. (2007). Estado não pode legislar sobre loteria, confirma STF. Disponível em: www.conjur.com.br/2007-mai-03/estado_nao_legislar_loteria_confirma_stf. Acesso em: 10 de janeiro de 2012.

Costa, Marcelo A. S. Lemos. (2003). Computação Forense. (2.ed., p.5-8). Campinas: Editora Millennium.

Dorea, Luiz E. C., Stumvoll, Victor P. & Quintela, Victor. (2003). Criminalística (2.ed). Campinas: Editora Millennium.

Dullien, Thomas & Rolles, Rolf. (2005). Graph-based Comparison of Executable Objects. Disponível em: citeseerx.ist.psu.edu/viewdoc/summary?doi=10.1.1.96.5076. Acesso em: 21 de novembro de 2011.

Eagle, Chris. (2008). The IDA Pro book: The unofficial guide to the world's most popular disassemble. San Francisco: No Starch Press.

Eilam, Eldad. (2005). Reversing: Secrets of Reverse Engineering. Indianapolis: Wiley.

Espindula, Alberi. (2006). Perícia Criminal e Cível: Uma visão geral para peritos e usuários da perícia (2.ed). Campinas: Millennium.

Espindula, Alberi. & Sousa, B. Galileu. (2011). Ciências Forenses - Uma introdução às principais áreas da Criminalística Moderna. Campinas: Millennium.

Farmer, Dan. & Venema, Wietse. (2007). Perícia Forense Computacional: Teoria e Prática Aplicada. São Paulo: Pearson Prentice Hall.

Fergotech. (2012). Descompactando arquivo UPX. 2012. Disponível em: fergonez.net/tutoriais/fergo/reveng/tut6/tut_engrev6.html. Acesso em: 24 de janeiro de 2012.

Flake, Halvar. (2004). Structural Comparison of Executable Objects. DIMVA. Disponível em: citeseerx.ist.psu.edu/viewdoc/summary?doi=10.1.1.83.6632. Acesso em: 21 de novembro de 2011.

Freitas, A. R. (2006). Perícia Forense Aplicada à Informática. Rio de Janeiro: Brasport.

G1. (2007). Videobingo. Disponível em: g1.globo.com/Noticias/Brasil/0,,MUL198148-5598, 00-RECEITA+FEDERAL+APREENDE+VIDEOBINGOS+EM+JOINVILLE.html. Acesso em: 25 de novembro de 2011.

G1. (2009). Videobingo. Disponível em: g1.globo.com/bomdiabrasil/0,,MUL1282310-16020,00-POLICIA+DESCOBRE+FABRICA+DE+CACANIQUEIS+NO+INTERIOR+DE+ SAO+PAULO.html. Acesso em: 25 de novembro de 2011.

G1. (2011a). Videobingo. Disponível em: g1.globo.com/sao-paulo/sao-jose-do-rio-preto-aracatuba/noticia/2011/12/bingo-funcionava-em-salao-de-beleza-de-fachada-em-rio-preto-sp. html. Acesso em: 14 de dezembro de 2011.

G1. (2011b). Videobingo. Disponível em: g1.globo.com/brasil/noticia/2011/12/operacao-tio-patinhas-apreende-108-maquinas-caca-niqueis-no-para.html. Acesso em: 31 de janeiro de 2012.

G1. (2012). Videobingo. Disponível em: g1.globo.com/parana/noticia/2012/01/policiais-civis-fecham-cassino-que-funcionava-em-mansao-em-curitiba.html. Acesso em: 31 de janeiro de 2011.

Hardware.com.br. (2007). Comprimindo executáveis com o UPX. Disponível em: www.hardware.com.br/dicas/comprimindo-executaveis-upx.html. Acesso em: 24 de janeiro de 2012.

Harris, Tom. (2011). Como funcionam os caça-níqueis. Disponível em: empresasefinancas. hsw.uol.com.br/caca-niqueis.htm. Acesso em: 15 de agosto de 2011.

ICT. (2011a). Products/Bill Acceptor. Disponível em: www.ict-america.com/product/bill_ acceptor.asp. Acesso em: 17 de agosto de 2011.

ICT. (2011b) Suppport/Support Guide. Disponível em: www.ict-america.com/pdffiles/BL-700%20Calibration%20Guide.pdf. Acesso em: 17 de agosto de 2011.

ICT. (2011c). Suppport/Manual Download (P70/P70P5). Disponível em: www.ictgroup. com.tw/files/p70&p70p5/P70-BRA6.P5(ID003).pdf. Acesso em: 17 de agosto de 2011.

Jurgens, David. (2011). Int 21: Dos Function Dispatcher. Disponível em: stanislavs.org/ helppc/int_21.html. Acesso em: 22 de dezembro de 2011.

Merian Webster. (2012). Slot machines. Disponível em: visual.merriam-webster.com/ sports-games/games/slot-machine_2.php. Acesso em: 7 de janeiro de 2012.

Michaelis. (2011a). Informática. Disponível em: michaelis.uol.com.br/moderno/portugues/ index.php?lingua=portugues-portugues. Acesso em: 18 de agosto de 2011.

Michaelis. (2011b). Jogo de azar. Disponível em: michaelis.uol.com.br/moderno/portugues /index.php?lingua=portugues-portugues. Acesso em: 18 de agosto de 2011.

Microsoft. (2012) FC. Disponível em: www.microsoft.com/resources/documentation/win dows/xp/all/proddocs/en-us/fc.mspx?mfr=true. Acesso em 22: de janeiro de 2012.

Narvaja, Ricardo (2005). Introducción al cracking con OllyDBG. Disponível em: Ricardo narvaja.info. Acesso em: 5 de setembro de 2011.

Nogueira, J. H. Matos. (2002) Máquinas Caça-Níqueis (a.4, n.12, p.18). Perícia Federal.

Paes, R. Santos (2008). Matemática e Raciocínio Lógico para Concursos e Vestibulares. 2.ed. Brasília: Vestcon.

Peters, James F. & Pedrycz, Witold. (2001). Engenharia de Software: Teoria e Prática (p.561-2). Rio de Janeiro: Campus.

Pirigov, Vlad. (2006). Disassembling Code: IDA Pro and SoftICE. Wayne: Alist.

Planalto. (1941). Lei das Contravenções Penais. Disponível em www.planalto.gov.br/ccivil_03/decreto-lei/Del3688.htm. Acesso em: 20 de agosto de 2011.

Portal Amazônia. (2011). Videobingo. Disponível em: www.portalamazonia.com.br/secao/noticias/rondonia/2011/09/01/policia-apreende-278-maquinas-caca-niqueis-em-porto-velho-em-2011. Acesso em: 31 de janeiro de 2012.

Pressman, Roger S. (2006). Engenharia de Software (6.ed). São Paulo: McGraw-Hill.

Reis, Albani B (2006). Metodologia Científica e Perícia Criminal. Campinas: Millennium.

Rodrigues, Jorílson S. (1999). Crimes por Computador (a.1, n.1, p.17). Perícia Federal.

SBC. (2011a). Currículo de Referência – CC e EC. Disponível em: www.sbc.org.br/index.php?option=com_jdownloads&Itemid=195&task=view.download&catid=36&cid=183. Acesso em: 17 de agosto de 2011.

SBC. (2011b). Currículo de Referência – SI. Disponível em: www.sbc.org.br/index.php?option=com_jdownloads&Itemid=195&task=view.download&catid=36&cid=185. Acesso em: 17 de agosto de 2011.

Schneier, Bruce. & Ferguson, Niels. (2003). Practical Cryptography. Indianapolis: Wiley.

Silva Júnior, A. Lopes (2007). A contravenção de exploração de jogo de azar. Disponível em: jus.com.br/revista/texto/10110/a-contravencao-de-exploracao-de-jogo-de-azar. Acesso em: 5 de janeiro de 2012.

Só Matemática. (2011). Probabilidade. Disponível em: www.somatematica.com.br/emedio/probabilidade.php. Acesso em: 20 de agosto de 2011.

Tocchetto, Domingos. & Espindula, Alberi. (2005). Criminalística: Procedimentos e Metodologias. Porto Alegre: Edição Própria.

Veloso, Fernando C. (2003). Informática: Conceitos Básicos (6.ed). Rio de Janeiro: Elsevier.

Vênere, Guilherme. (2009). Engenharia Reversa de Código Malicioso. São Paulo: Escola Superior de Redes/Rede Nacional de Pesquisa.

Youtube. (2010a). Calibração P70. Disponível em: www.youtube.com/watch?v=1gAHm2P6-N0. Acesso em 23 de junho de 2011.

Youtube (2010b). Fim Calibração P70.avi. Disponível em: www.youtube.com/watch?v=Jk1u7CwPW3E. Acesso em 23 de junho de 2011.

Zelenovsky, Ricardo. & Mendonça, Alexandre. (2006). PC: Um Guia Prático de Hardware e Interfaceamento. Rio de Janeiro: MZ Editora.

ANEXOS

A – LIGAÇÃO DOS EMULADORES DE TECLADO

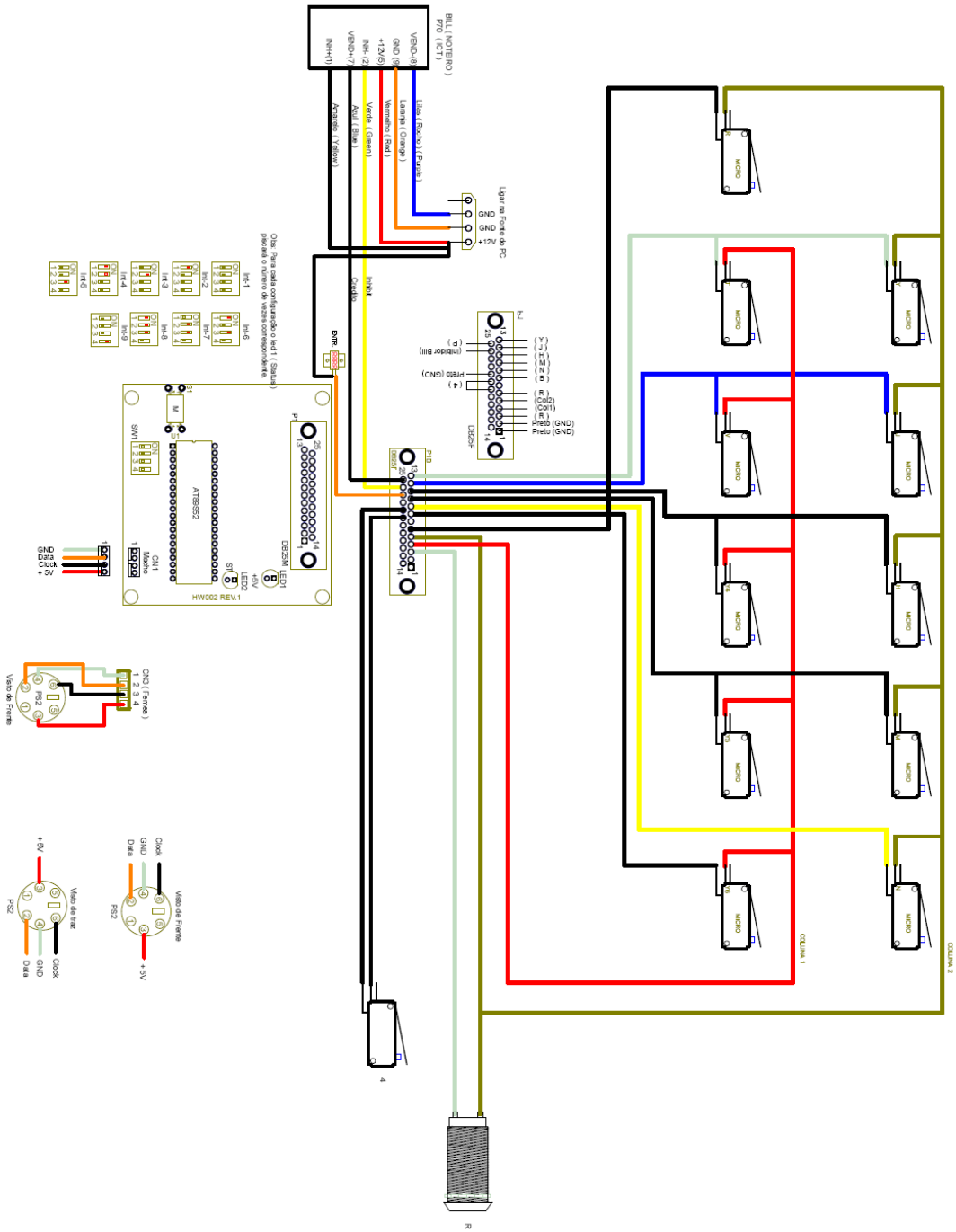

Figura A.1. Esquema de conexão do emulador de teclado HW002 com 10 botões.

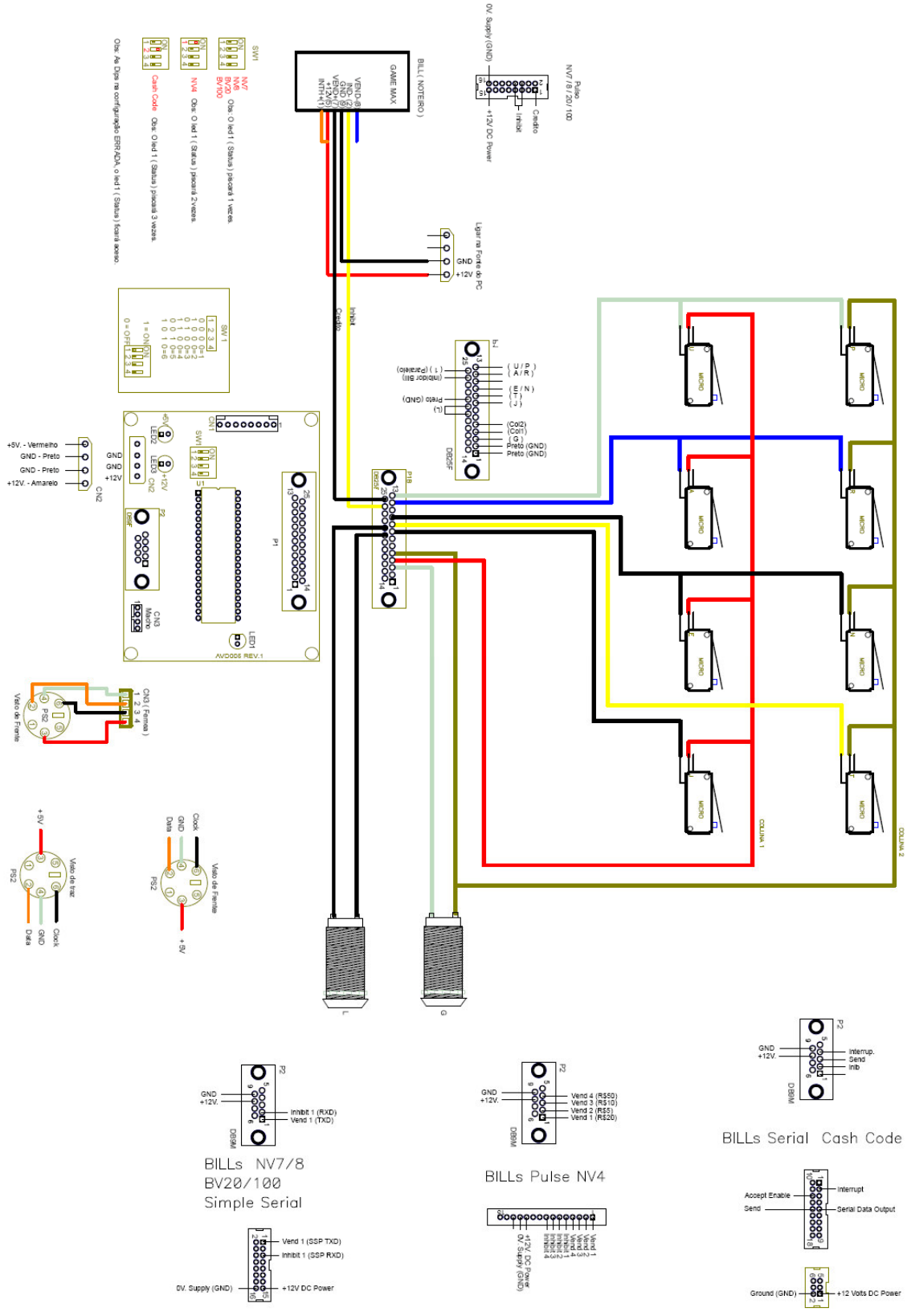

Figura A.2. Esquema de conexão do emulador de teclado AVD005 com 8 botões.