UNIVERSIDADE FEDERAL DE SERGIPE

CENTRO DE CIÊNCIAS EXATAS E TECNOLOGIA

PROGRAMA DE PÓS-GRADUAÇÃO EM CIÊNCIA DA COMPUTAÇÃO

Uma Abordagem para Análise de Desempenho e Eficiência Energética em Dispositivos Embarcados com uso do Asterisk

Dissertação de Mestrado

Adauto Cavalcante Menezes

São Cristóvão – Sergipe

2018

UNIVERSIDADE FEDERAL DE SERGIPE

CENTRO DE CIÊNCIAS EXATAS E TECNOLOGIA

PROGRAMA DE PÓS-GRADUAÇÃO EM CIÊNCIA DA COMPUTAÇÃO

Adauto Cavalcante Menezes

Uma Abordagem para Análise de Desempenho e Eficiência Energética em Dispositivos Embarcados com uso do Asterisk

Dissertação de Mestrado apresentada ao Programa de Pós-Graduação em Ciência da Computação da Universidade Federal de Sergipe como requisito parcial para a obtenção do título de mestre em Ciência da Computação.

Orientador(a): Admilson de Ribamar Lima Ribeiro
Coorientador(a): Edward David Moreno Ordonez

São Cristóvão – Sergipe

2018

Menezes, Adauto Cavalcante
A543a Uma abordagem para análise de desempenho e eficiência energética em dispositivos embarcados com uso do Asterisk / Adauto Cavalcante Menezes ; orientador Admilson de Ribamar Lima Ribeiro. - São Cristóvão, 2018.
91 f. : il.

Dissertação (Mestrado em Ciência da Computação) - Universidade Federal de Sergipe, 2018.

1. Asterisk (Programa de computador). 2. Sistemas embarcados (Computação). 3. Processamento eletrônico de dados. I. Ribeiro, Admilson de Ribamar Lima orient. II. Título.

CDU 004.41

Adauto Cavalcante Menezes

Uma Abordagem para Análise de Desempenho e Eficiência Energética em Dispositivos Embarcados com uso do Asterisk

Dissertação de Mestrado apresentada ao Programa de Pós-Graduação em Ciência da Computação da Universidade Federal de Sergipe como requisito parcial para a obtenção do título de mestre em Ciência da Computação.

Trabalho aprovado. São Cristóvão – Sergipe, 09 de Abril de 2018:

Admilson de Ribamar Lima Ribeiro
Orientador

Edward David Moreno Ordonez
Coorientador

Ricardo José Paiva de Britto Salgueiro
Interno a instituição

Carlos Humberto Llanos Quintero
Externo a instituição

São Cristóvão – Sergipe
2018

Dedico esta dissertação de Mestrado a toda a minha família, minha noiva, amigos e professores que me deram o apoio necessário para chegar até aqui. Em memória de Francisco Adalberto Menezes, meu saudoso pai.

Resumo

A comunicação por voz sobre IP (VoIP) dominará o mundo da computação nos próximos anos. Para realizar a transmissão VoIP, faz-se necessário a codificação e decodificação da voz. Esse processo consome os principais recursos computacionais, como o processador e memória. As indústrias das telecomunicações disponibilizam equipamentos com valores elevados, o que torna o acesso a essa tecnologia ainda bastante restrito. Dispositivos embarcados executam sistemas com alto complexo de criticidade. O *software* livre de comunicação por voz sobre IP Asterisk tem como principal função implementar todas as funcionalidades de uma central telefônica. Trata-se de uma alternativa viável para utilizar em dispositivos embarcados. Essas tecnologias prometem reduzir custos e maximizar resultados. Esta dissertação realiza uma análise de desempenho em três dispositivos embarcados modernos (Raspberry Pi 3, Orange Pi Plus 2 e Banana Pi M3), com uso do sistema de comunicação por voz sobre IP Asterisk. A análise de desempenho consiste em aferir o *jitter*, *delay* e vazão, assim como o quantitativo de chamadas simultâneas suportadas em cada dispositivo com os protocolos SIP e IAX2 com os CODEC'S G.711a, G.711u, Gsm, Speex, Ilbc, G.722, e em paralelo, monitorar o consumo de memória RAM, processamento e energia. Os resultados mostraram que os dispositivos Raspberry Pi 3 e Banana Pi M3 suportam de forma satisfatória aproximadamente 300 chamadas simultâneas com consumo de memória, processamento e energia moderado. Todavia, o dispositivo Orange Pi Plus 2 não obteve desempenho satisfatório no consumo de processamento.

Palavras-chave: Asterisk, Sistemas Embarcados, Análise de Desempenho, Eficiência Energética.

Abstract

Voice over IP (VoIP) communication will dominate the computing world for years to come. To perform VoIP transmission, it is necessary to encode and decode the voice. This process consumes the major computing resources, such as processor and memory. The telecommunication industries provide equipment with high values, which makes access to this technology still very restricted. Embedded devices run systems with high criticality complex. Free voice over IP communication software Asterisk has as main function to implement all the functionalities of a telephone exchange. This is a viable alternative to use on embedded devices. These technologies promise to reduce costs and maximize results. This dissertation performs a performance analysis on three modern embedded devices (Raspberry Pi 3, Orange Pi Plus 2 and Banana Pi M3) using the Asterisk voice over IP communication system. The performance analysis consists in measuring the jitter, delay and bandwidth, as well as the quantitative simultaneous calls supported in each device with the SIP and IAX2 protocols with CODEC's G.711a, G.711u, Gsm, Speex, Ilbc, G. 722, and in parallel, monitor the consumption of RAM, processing and power. The results show that the Raspberry Pi 3 and Banana Pi M3 devices satisfactorily support approximately 300 concurrent calls with memory consumption, processing, and moderate power. However, the Orange Pi Plus 2 device did not perform satisfactorily in the processing consumption.

Keywords: Asterisk, Embedded Systems, Performance Analysis, Energy Efficiency.

Lista de ilustrações

Lista de tabelas

Lista de abreviaturas e siglas

AES	Advanced Encryption Standard
CPU	Unidade Central de Processamento
GPL	GNU General Public License
IAX	Inter Asterisk Exchange
IAX2	Inter Asterisk Exchange version 2
IETF	Internet Engineering Task Force
IP	Internet Protocolo
MOS	Mean Opinion Score
NAT	Network Address Translation
OSI	Open Systems Interconnect
CVS	Concurrent Version System
PBX	Private Branch Exchange
QoS	Qualidade de serviço
RTP	Real Time Protocol
SDP	Session Description Protocol
SER	Sip Express Router
SIP	Session Initiation Protocol
TDM	Time Division Multiple
UDP	User Datagram Protocol
VAD	Supressão de licencio
VoIP	Voice over Internet Protocol
Wi-Fi	Wireless Fidelity

Sumário

1 Introdução

O termo *Voice Over Internet Protocol* (VoIP) é conceituado como a comunicação de voz em redes que utilizam o *Internet Protocol* (IP), o qual desenvolveu-se com o aparecimento da Telefonia IP, visto que consiste no fornecimento de serviços de telefonia com utilização da rede IP para o estabelecimento de chamadas e comunicação de voz (BERNAL, 2007). Em meados do ano de 1990, consolidou-se a definição do VoIP, quando surgiu o *Internet Phone* da VocalTec Communications: o primeiro software comercial que possibilitava a comunicação de voz sobre IP, porém com qualidade de comunicação precária (COLCHER et al., 2005).

Segundo Androulidakis (2016), as centrais telefônicas privadas servem para realizar comunicação entre telefones internos e a rede pública telefônica. Os sistemas de comunicação podem ser de dois tipos: IP, ou também denominado Private Branch Exchange (PBX) IP, e sistemas de comunicação com multiplexação por divisão de tempo (TDM), ou PBX TDM convencional. Seu *software* pode ser oferecido de modo proprietário ou através de código aberto. A comunicação com o meio externo depende da interface para interligar as linhas analógicas ou digitais, geralmente fornecidas por operadoras de telecomunicações. Aquela que é realizada por meio dos telefones internos depende exclusivamente da própria central. Inicialmente, a principal vantagem do uso dos sistemas de comunicação PBX foi o custo na chamada das linhas internas, pois ocorre uma comutação de circuitos interna, o que torna a chamada gratuita.

Ainda de acordo com Androulidakis (2016), os sistemas de comunicação por voz ganharam popularidade, passaram a ter funcionalidades e serviços que não estavam disponíveis nas operadoras de telecomunicações, tais como grupos de busca, encaminhamento de chamadas e discagem por ramal. Segundo Sulkin (2002), a tendência nos últimos anos é migrar para telefonia IP.

No ano de 1999, surgiu o *software* para comunicação de voz sobre IP Asterisk. Segundo Bryant, Madsen e Meggelen (2013), Asterisk é um *software* livre que executa todas as funções de uma central telefônica convencional. Ele foi desenvolvido pela empresa Digium, a qual, além

de manter e fornecer o *software* Asterisk, investe também em *hardware* para a área de telefonia. Atualmente recebe inúmeras contribuições de desenvolvedores de todo o mundo, por ser uma área de aplicação promissora e em constante desenvolvimento.

Todavia, é necessário prover um mecanismo para reduzir despesas na área de telefonia, bem como ampliar e incentivar o conhecimento de alunos e pesquisadores da área de tecnologia da informação e telecomunicações. Nesse contexto, justifica-se a importância pela eficácia e redução de custos da nova geração de sistemas personalizados, revolucionários para comunicação de voz sobre IP. Dessa forma, reforça-se a ideia de que qualquer avanço de qualidade nessa tecnologia pode propiciar um aumento considerável na qualidade da comunicação por voz, tanto em ambiente acadêmico quanto na indústria das telecomunicações. Assim, têm-se a motivação para a realização deste trabalho.

1.1 Problemática e Hipótese

A utilização da Internet como meio de comunicação por voz aumenta constantemente e torna-se uma opção econômica quando comparada à telefonia convencional, principalmente nas chamadas de longas distâncias.

Para isso, utilizam-se sistemas de comunicação por voz sobre IP. Dentre os quais, é possível destacar o Asterisk: um *software* para comunicação VoIP, que utiliza um conceito de *software* livre (GPL). O Asterisk funciona em plataforma Unix, bem como em plataformas Linux (GONÇALVES, 2007). Segundo Voip-info (2017), o Asterisk utiliza SIP e IAX como principais protocolos para realizar a comunicação entre dispositivos, seja comunicação interna ou comunicação externa.

Abid et al. (2012) demonstraram que é possível baixar o custo com a implementação de um sistema de comunicação de voz sobre IP. Para isso utilizam o *software* Asterisk embarcado em um dispositivo de baixo custo. Já Edan et al. (2016), realizaram uma avaliação de desempenho de Qos com uso dos protocolos SIP e IAX, e para tanto, utilizaram um *software* gratuito baseado em Asterisk.

Nesse contexto, é necessário prover um mecanismo para reduzir despesas na área de telefonia, de preferência com características similares a de uma central telefônica convencional. É possível, a partir do Asterisk, implementar um sistema de comunicação por voz em dispositivos embarcados e de baixo custo que forneça os mecanismos necessários de uma central telefônica convencional com um baixo consumo de energia, processamento e memória, assim como suporte a um elevado número de chamadas simultâneas.

1.2 Objetivos

O objetivo deste trabalho é implementar e realizar uma análise de desempenho e eficiência energética em dispositivos embarcados com o *software* de comunicação por voz sobre IP Asterisk, a fim de verificar o comportamento do mesmo em diferentes dispositivos embarcados, bem como o consumo de energia em momentos de elevada utilização do sistema. Para alcançar o objetivo do trabalho, é necessário buscar alguns objetivos específicos.

Os objetivos específicos desse trabalho são listados a seguir:

- Identificar o consumo da vazão, *jitter* e *delay* em diferentes dispositivos embarcados.
- Comparar o desempenho de chamadas com o protocolo SIP e IAX.
- Implementar gerador de cargas SIP e IAX.
- Investigar o quantitativo de chamadas simultâneas suportadas em cada dispositivo.
- Analisar o consumo energético durante elevada utilização do sistema.
- Verificar o consumo de memória e processamento no decorrer de elevados fluxos de chamadas.
- Analisar o desempenho dos 3 dispositivos embarcados.

1.3 Justificativa

Como apresentado na introdução deste trabalho, sistemas de comunicação por voz são imprescindíveis para empresas, indústrias e instituições de ensino, além de grandes e pequenas corporações. Dessa forma, diante da natureza do ambiente, o Asterisk se mostra como um conceito importante para viabilizar uma forma de comunicação de baixo custo com uso de dispositivos embarcados. Dessa maneira, a indústria alavanca grandes projetos proprietários, e assim eleva arbitrariamente o custo ao consumidor final.

Alguns trabalhos relacionados utilizam o Asterisk como um sistema de comunicação por voz. Inclusive, o mesmo é aplicado no modelo proposto por Abid et al. (2012). Porém, o seu trabalho não contempla a fase de simulação de chamadas simultâneas e medições de eficiência energética. Edan et al. (2016), realizaram a avaliação de desempenho dos protocolos SIP e IAX2, porém não utilizam dispositivos embarcados e não realizam medições de eficiência energética e processamento.

Os trabalhos iniciados pelos autores supracitados deixaram sugestões para trabalhos futuros, como por exemplo, a realização dos mesmos testes com uso de chamadas simultâneas, assim como embarcar o Asterisk em outras plataformas. Dessa forma, é possível ampliar o

horizonte da pesquisa e, por este motivo, será dado continuidade em relação à medição em ambiente real e uma sequência de testes mais apurados, como uma análise de desempenho e eficiência energética.

1.3.1 Organização da Dissertação

Para facilitar a navegação e melhor entendimento, este documento está estruturado em seis capítulos, que são:

- Capítulo 1 - Introdução
- Capítulo 2 - Fundamentação Teórica
- Capítulo 3 - Trabalhos Correlatos
- Capítulo 4 - Metodologia para Prototipação e Coleta de Dados
- Capítulo 5 - Experimentos e Resultados
- Capítulo 6 - Conclusão

Na Introdução, apresentamos o problema, a justificativa e o que foi proposto na dissertação. No capítulo Fundamentação Teórica, são abordados os principais temas do trabalho e tecnologias a fim de contextualizar o leitor com foco em VoIP, Asterisk, arquitetura do Asterisk, protocolos SIP, IAX e dispositivos embarcados. No capítulo Trabalhos Correlatos, são apresentados os trabalhos relacionados ao problema descrito, suas contribuições e limitações.

O capítulo Metodologia para Prototipação e Coleta de Dados descreve a metodologia do trabalho de forma detalhada. No capítulo Experimentos e Resultados, é apresentado o cenário das medições e os dados utilizados para a coleta dos resultados. Por fim, no capítulo de Conclusão são abordadas as considerações finais da pesquisa e trabalhos futuros.

2

Fundamentação Teórica

Neste capítulo há um tópico para cada um dos assuntos que viabilizaram a contextualização do trabalho realizado. Os tópicos presente neste capítulo correspondem respectivamente à: VoIP, Asterisk, Arquitetura do Asterisk, Protocolo SIP, Protocolo IAX e Dispositivos Embarcados.

2.1 VoIP

Diante do crescente desenvolvimento da Internet, surgiram diversas novas tecnologias. O termo VoIP em português significa Voz sobre IP, trata-se de uma tecnologia que possibilita realizar chamadas telefônicas via rede de computadores e até via Internet. Para o usuário final o uso desta tecnologia é transparente, é o provedor de serviços que gerencia o transporte da comunicação (KELLY, 2005).

Para ocorrer a comunicação entre os usuários da tecnologia VoIP, se faz necessário a digitalização da voz, a qual realiza a conversão do áudio analógico para o digital. Esta conversão é realizada por codificadores e decodificadores, denominados CODECS, proveniente do termo em inglês “encode” e “decode”. As aplicações do serviço de comunicação de voz sobre IP, em sua maioria, suportam mais de um CODEC, assim é possível negociar qual CODEC utilizar na comunicação.

Segundo Keller (2011), a escolha do CODEC adequado deve-se basear nas necessidades do seu ambiente, como a capacidade de processamento, disponibilidade de banda e quantidade de chamadas simultâneas. Na tecnologia VoIP, existe um processo denominado de transcodificação, que consiste na conversão entre CODECS. Além disso, os CODECS influenciam na qualidade da voz na chamada. Keller (2011), cita como principais características dos CODECS a taxa de *bits*, o intervalo de amostra, o tamanho de amostra e o tamanho de *payload* de voz, bem como aborda os CODECS G.711, G.729a, G.722, GSM, ILBC, SPEEX como os principais.

Com a digitalização da voz, a qualidade do áudio é fundamental. Assim, faz-se necessário preocupar-se com o atraso no envio dos pacotes (ou no termo em inglês frequentemente utilizado "delay"), perda de pacotes, *jitter*, eco, supressão de silêncio (VAD), *Mean Opinion Score* (MOS) e qualidade de serviço (QoS) (HARTPENCE, 2013).

Para realizar uma comunicação VoIP, utiliza-se protocolos a fim de iniciar, estabelecer e finalizar a comunicação. Os protocolos *Session Initiation Protocol* (SIP) e H.323, são os mais comuns, já o *Real-time Transport Protocol* (RTP) realiza o transporte e negociação da mídia. Esses protocolos possibilitam uma comunicação multimídia (DEON, 2010).

A tecnologia VoIP possibilita uma série de benefícios, entre eles: a redução de custos; utilização de uma única infraestrutura de rede; mobilidade; o controle do sistema de comunicação; e novas funcionalidades não disponíveis nos sistemas de comunicação tradicionais (OUAKIL; PUJOLLE, 2008).

2.2 Asterisk

Asterisk é um *software* livre e de código aberto, que surgiu no final da década de 1990. Sua primeira versão foi publicada por Mark Spencer, em cinco de dezembro de 1999. Na mesma época surgiram *softwares* livres para telecomunicações, desenvolvidos em torno dos protocolos H.323 e SIP, como OpenH323 e SIP *Express Router* (SER), porém ambos descontinuados (SULTAN, 2009).

Ao contrário da maioria dos *softwares* livres na área de telecomunicações, o Asterisk não é resultado do trabalho de um ou mais desenvolvedores que elaboram a partir de um artigo, a fim de descrever um protocolo padrão. O Asterisk foi desenvolvido a partir da necessidade pragmática do jovem gerente de uma empresa prestadora de serviços de suporte em Linux e *softwares* livres, chamado Mark Spencer. Ele desejava melhorar a eficácia do suporte técnico, a fim de proporcionar a possibilidade dos clientes deixarem mensagens telefônicas de forma a orientá-los para o atendimento técnico necessário (SULTAN, 2009).

Segundo Yeung (2015), o fato de o *software* Asterisk ser gratuito o deixou em grande relevância. Também há vários servidores baseados em Asterisk que podem ser obtidos gratuitamente, por exemplo, FreePBX, Elastix, Trixbox, FreeSwitch.

Ainda de acordo com Yeung (2015), o suporte a estes *softwares* se dá através de fóruns cujos membros publicam uma série de soluções para a maioria dos problemas. Pode-se encontrar informações sobre a configuração e operação. Também é possível encontrar serviços de suporte pagos, isso significa que os usuários não são abandonados quando enfrentam problemas técnicos.

Keller (2011) considera o Asterisk uma central telefônica híbrida, pois tanto implementa os protocolos VoIP quanto as funções de uma central telefônica convencional. O Asterisk é a implementação de uma central telefônica para comutação de ramais privados ou PBX em

software. Ele é distribuído de forma gratuita através da empresa Digium, que segue um padrão de licença GNU *General Public License* (GPL). O nome Asterisk vem do símbolo ‘ * ’, muito comum no mundo da telefonia.

Para Mahler (2005), a principal função do Asterisk é a de uma central telefônica. Trata-se de um *software* que faz o gerenciamento das chamadas telefônicas, normalmente utilizado em pequenas, médias e grandes corporações. Comuta chamadas telefônicas para ramais diretamente conectados ou encaminha as chamadas através da rede de telefonia pública.

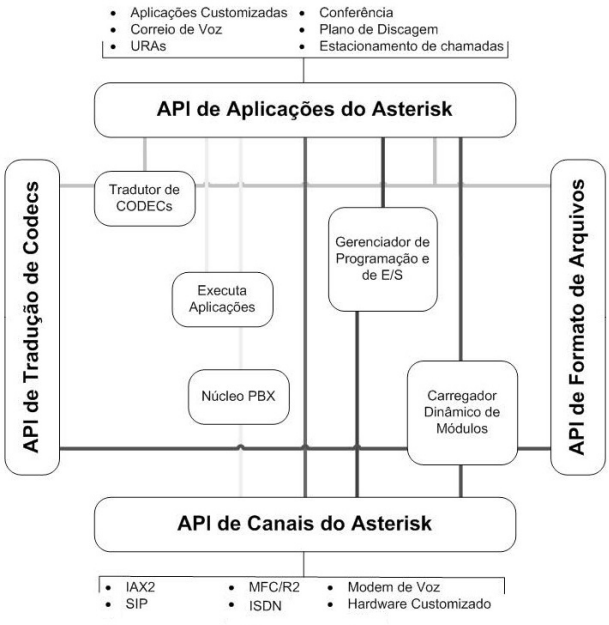
O *software* Asterisk inclui muitos recursos disponíveis em sistemas de comunicação por voz, como: correio de voz, chamada em conferência, resposta de voz interativa (menus do telefone) e distribuição automática de chamadas. Por esse motivo, ele se torna ainda mais competitivo do que as centrais telefônicas convencionais.

Desde o surgimento da tecnologia VoIP, que transfere o serviço de telefonia convencional para telefonia sobre IP e o integra a uma rede de computadores, os termos PBX, PBX IP, IP-PBX surgiram, e isso reflete o fato de que o PBX agora vem como um *software*, em vez de um equipamento especializado (MAHLER, 2005). Todavia, entende-se por PBX, como um sistema de comunicação por voz.

2.2.1 Arquitetura do Asterisk

Segundo Keller (2011), o funcionamento e a operação do Asterisk se dão basicamente em quatro módulos, dentre eles estão: o protocolo, canal de comunicação, CODECS e aplicação. A figura 1, mostra sua arquitetura.

Figura 1 – Arquitetura do Asterisk.



Fonte: Adaptado de Keller (2011).

O protocolo é responsável em prover a comunicação com o sistema, a qual geralmente se dá através dos protocolos SIP ou *Inter Asterisk Exchange* (IAX). Dessa forma, o servidor encaminha a sinalização e o transporte de mídia entre os integrantes da conversação. O Canal de comunicação dialoga-se com ambientes externos e internos, sejam *hardwares* ou outros *softwares* para comunicação. Já o CODEC é a forma como o áudio é digitalizado. Por último, o módulo de aplicação responsável por tratar as chamadas realiza praticamente todas as funcionalidades do Asterisk, seja ao fazer ou receber chamadas (GONÇALVES, 2007).

De acordo com Mahler (2005), quando o Asterisk inicializa, a leitura do carregador dinâmico de módulos ocorre e os *drivers* são ativados, estes possibilitam o funcionamento de diversos serviços de aplicações. O núcleo, ou *core*, aceita as chamadas telefônicas a partir das interfaces. Desse modo, as chamadas são tratadas de acordo com as informações encontradas no plano de discagem. Há também um gerenciador de programação e de E/S, este fica à disposição dos *drivers* e aplicações. Além disso, o Tradutor de CODECS realiza a compressão dos diferentes tipos de CODECS.

Bryant, Madsen e Meggelen (2013), considera o Asterisk como um sistema que comporta várias tecnologias distintas. Além disso, o conceitua como um sistema de comunicação de voz sobre IP bem projetado. Assim, é possível proporcionar um notável equilíbrio entre flexibilidade e complexidade.

2.2.2 Protocolo SIP

A primeira versão do protocolo SIP surgiu em 1997, diante do crescente desenvolvimento do mundo das telecomunicações. Esse protocolo foi criado com o propósito de se adaptar à Internet, particularmente para a rede suportar um maior número de usuários. Trata-se de uma abordagem escalável e flexível, que possibilita adicionar novos recursos a qualquer momento, sem a necessidade de alterar os componentes existentes (OUAKIL; PUJOLLE, 2008).

Para Flanagan (2012), o SIP é um protocolo de sinalização, responsável por realizar a conexão, estabelecimento e finalização da comunicação. Esse processo ocorre entre uma comunicação ponto a ponto. Padronizado através da RFC3261 por *Internet Engineering Task Force* (IETF), o protocolo SIP possui semelhanças com os protocolos HTTP e SMTP, devido a sua troca de mensagens em formato de texto.

De acordo com Ouakil e Pujolle (2008), o protocolo SIP não assegura o transporte dos dados, apenas estabelece a comunicação entre os interlocutores e garante sinalização. Localizado na camada de aplicação do modelo de referência *Open Systems Interconnect* (OSI), ele opera em uma arquitetura cliente-servidor. Segundo Keller (2011), o processo de sinalização é realizado através do protocolo *User Datagram Protocol* (UDP) na porta 5060. Já o *Real-time Transport Protocol* (RTP) realiza a negociação e transporte da mídia através do protocolo, também UDP, e porta de 10000 a 20000.

O protocolo SIP realiza a troca de informações entre os clientes e servidores através de solicitações e das respostas a essas solicitações. Assim, quando uma chamada é realizada, os comandos de sinalização *register, invite, ack, options, subscribe, notify, cancel e bye* podem ser utilizados. Além disso, a cada solicitação realizada há uma série de possíveis respostas, tais como: informacional, redirecionada, erro no cliente, erro no servidor e falha global (SINNREICH; JOHNSTON, 2006).

Ainda de acordo com Sinnreich e Johnston (2006), ao iniciar uma sessão, o protocolo *Session Description Protocol* (SDP) estabelece a descrição das informações imprescindíveis para o transporte da mídia através da rede. Esse protocolo provê qual o tipo de mídia a ser usado, bem como as informações necessárias para realizar a configuração dos dados.

Segundo Johnston (2009), o SIP é um protocolo em frequente evolução; novas extensões e aplicações estão em desenvolvimento. Porém, o SIP precisa melhorar continuamente a sua interoperabilidade, de forma a prover melhores serviços para a comunicação ponto a ponto e multiponto.

2.2.3 Protocolo IAX

O projeto Asterisk gerou um segundo projeto, o *Inter-Asterisk-Exchange* (ou IAX), desenvolvido com o objetivo de possibilitar a comunicação entre os servidores Asterisk. Trata-se de um protocolo aberto que realiza o transporte de sinalização e da mídia. Ele utiliza a porta UDP/IP 4569, o que leva a entender que sua configuração seja mais simples do que outros protocolos. Em 2009, o IETF realizou a padronização e homologação do *Inter-Asterisk-Exchange* 2 (IAX2) através da RFC5456 (KELLER, 2011).

Ainda de acordo com Keller (2011), o protocolo IAX2 tem como base de funcionamento a transmissão da informação de duas formas: *frames* completos e *mini frames*. Os *frames* IAX completos são geralmente utilizados para transmitir mensagens de sinalização e informações sobre mídia. Porém, não é recomendado, uma vez que *frames* dedicados devem ser utilizados para o transporte de informações de mídia. Além disso, os *frames* IAX completos são utilizados para troca de mensagens de forma confiável, através do envio de uma mensagem de confirmação que deve ser enviada através do participante remoto. Dessa forma, confirma-se que o pacote completo foi recebido e processado (Mohamed Boucadair, 2009).

A denominação de *mini frames* foi dada devido ao cabeçalho reduzido para quatro octetos. Os *mini frames* são utilizados apenas para o envio da voz, assim que cada sessão é estabelecida. Diferente dos *frames* completos, os *mini frames* não exigem confirmação do participante remoto (Mohamed Boucadair, 2009).

O protocolo IAX permite a troca de chaves compartilhadas, tanto pode ser utilizado com texto simples ou em conjunto com mecanismos de criptografia como *Advanced Encryption Standard* (AES), que se baseiam em um segredo compartilhado (RFC5456, 2016).

A forma de autenticação do IAX é implementada através de uma troca de pedidos de autenticação que encerram um desafio de segurança. Este desafio de autenticação deve ser respondido através do servidor remoto e criptografado de acordo com o método de criptografia adotado. Se a negociação de criptografia falhar, a chamada deve ser encerrada (RFC5456, 2016).

A filosofia proposta através do protocolo IAX difere em um aspecto importante: apresenta-se transparente aos *gateways*, o que facilita o seu uso em ambientes com *Network Address Translation* (NAT) e *firewalls*. Ao contrário do protocolo SIP, que realiza apenas a função de sinalização e o transporte do fluxo se dá via RTP. O protocolo IAX é tanto um protocolo de transporte quanto um protocolo de sinalização (Mohamed Boucadair, 2009).

Com o uso do protocolo IAX, nota-se uma redução no uso de largura de banda. O protocolo SIP foi projetado para uso de fluxos de multimídia em geral, já o protocolo IAX foi projetado especificamente para o problema de transporte e sinalização da voz. Dessa forma, descarta as considerações de aplicações multimídia. Assim, o protocolo IAX corresponde a objetivos simples e bem definidos (GONÇALVES, 2007).

2.3 Dispositivos Embarcados

Nos dias atuais, os dispositivos embarcados estão se tornando mais populares. Eles estão presentes em nossa vida e em diferentes setores da indústria. Aparelhos de consumo simples, tais como fornos de micro-ondas, máquinas de lavar, frigoríficos e assim por diante, contemplam vários micro controladores para executar um conjunto predefinido de funções e procedimentos (BERTOLOTTI; HU, 2015).

De acordo com Dey e Mukherjee (2016), dispositivos embarcados, em sua maioria, executam sistemas com alto complexo de criticidade. Grande parte desses sistemas são baseados em sensores e atuadores. Um sensor verifica o comportamento do mundo exterior e envia as informações para um micro controlador integrado, o que possibilita gerar uma determinada ação, a depender da programação do sistema.

Para White (2011), um dispositivo embarcado é um dispositivo informatizado, construído propositadamente para a sua aplicação. Atualmente, os dispositivos embarcados podem ser programados em linguagens de alto nível e possuem sistemas operacionais. Os dispositivos embarcados são ideais do ponto de vista econômico para prover um conjunto limitado de serviços a automóveis, aeronaves e centrais de comutação de telecomunicações (SIEWERT; PRATT, 2016).

2.3.1 *Single-Board Computers*

Atualmente existe uma grande oferta de dispositivos embarcados no mercado, também denominados SBC (*Single-Board Computers*). Como exemplo, é possível citar Raspberry Pi

Zero, Raspberry 2 Model B, Raspberry Pi 3 Model B, entre outros. A tabela 1 ilustra uma variedade de dispositivos embarcados, suas configurações e preço.

Tabela 1 – Embarcados

Fabricante	**Raspberry Pi**	**Raspberry Pi**	**Raspberry Pi**	**Banana Pi**	**Banana Pi**	**Orange Pi**	**Orange Pi**	**Orange Pi**	**Orange Pi**	**Odroid**	**Odroid**	**Odroid**
Modelo	Zero	2 Model B	3 Model B	M2	M3	One	PC	Plus	Plus 2	C1+	C2	XU4
CPU Cores	1	4	4	4	8	4	4	4	4	4	4	4+4
CPU Freq.	1GHz	0.9GHz	1.2GHz	1GHz	2GHz	1.5GHz	1.5GHz	1.5GHz	1.5GHz	1.5GHz	2GHz	2.1GHz 1.5GHz
Memoria	512MB DDR2	1GB DDR2	1GB DDR2	1GB DDR2	2GB DDR3	512MB DDR3	1GB DDR3	1GB DDR3	2GB DDR3	1GB DDR3	2GB DDR3	2GB DDR3
Armazanamento	MicroSD	MicroSD	MicroSD	MicroSD	MicroSD USB Sata 2.0	MicroSD	MicroSD	MicroSD USB Sata 2.0	MicroSD USB Sata 2.0	MicroSD eMMC	MicroSD eMMC	MicroSD eMMC
Armazanamento OnBoard	não	não	não	não	8GB eMMC	não	não	8GB eMMC	16GB eMMC	não	não	não
USB	1 OTG	4	4	4 + 1 OTG	2 + 1 OTG	1 + 1 OTG	3 + 1 OTG	4 + 1 OTG	4 + 1 OTG	4 + 1 OTG	4 + 1 OTG	1
HDMI	Sim	Sim	Sim	Sim	Sim	Sim	Sim	Sim	Sim	Sim	Sim	Sim
Alimentação	5V 1A	5V 2A	5V 2.5A	5V 2A	5V 2A	5V 2A	5V 2A	5V 2A	5V 2A	5V 2A	5V 2A	5V 4A
Suporte Linux	Sim	Sim	Sim	Sim	Sim	Sim	Sim	Sim	Sim	Não	Não	Sim
Preço	$25	$35	$40	$49	$59,99	$20	$24,99	$30	$59,99	$35	$46	$59

Fonte: Adaptado de Loverpi (2017).

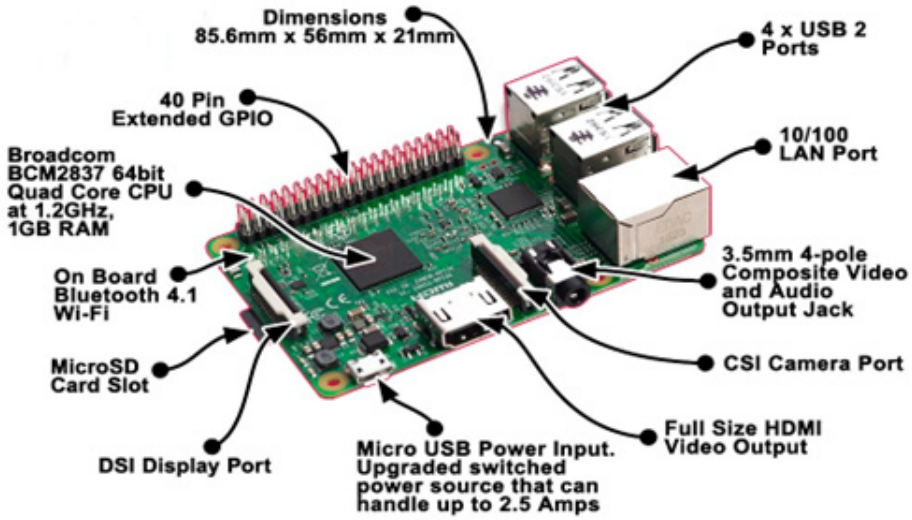
Segundo a Digium (2017), não é fácil dimensionar um *hardware* mínimo para a instalação do sistema de comunicação por voz sobre IP Asterisk. Todavia, para essa difícil questão não há uma resposta precisa. Recomenda-se utilizar sempre um *hardware* um pouco além das necessidades. Sugere-se ainda que caso o sistema necessite de mais de 20 chamadas simultâneas, deve-se utilizar um servidor dedicado. Assim, para a realização deste trabalho, foram escolhidos três dispositivos embarcados de baixo custo e com boas configurações, são eles: Raspberry Pi 3, Banana Pi M3 e Orange Pi Plus 2.

2.3.1.1 Raspberry Pi 3

Trata-se de um dispositivo embarcado que possui um poder de processamento considerável, devido ao processador Broadcom BCM2837 de 64 *bits* e *clock* de 1.2GHz. Com *Wifi* e *Bluetooth* 4.1 integrados, evita que o usuário compre adaptadores adicionais, assim deixa as portas USB livres para serem utilizadas por outras aplicações, como por exemplo, a conexão de periféricos externos.

A placa ainda possui 1G de memória RAM, adaptador para cartão microSD e GPU Videocore IV 3D. É uma placa que possui compatibilidade com o modelo anterior, Raspberry Pi 2, não só em termos de aplicações como também em seu *layout*, já que os seus conectores foram mantidos na mesma posição, assim como o tamanho e a perfuração da placa. Com a Raspberry Pi 3, é possível executar diversas distribuições Linux como o Raspbian e Ubuntu, além do Windows 10 IoT. A figura 2 ilustra a placa Raspberry Pi 3.

Figura 2 – Raspberry Pi 3.



Fonte: Adaptado de Zapals (2017).

2.3.1.2 Banana Pi M3

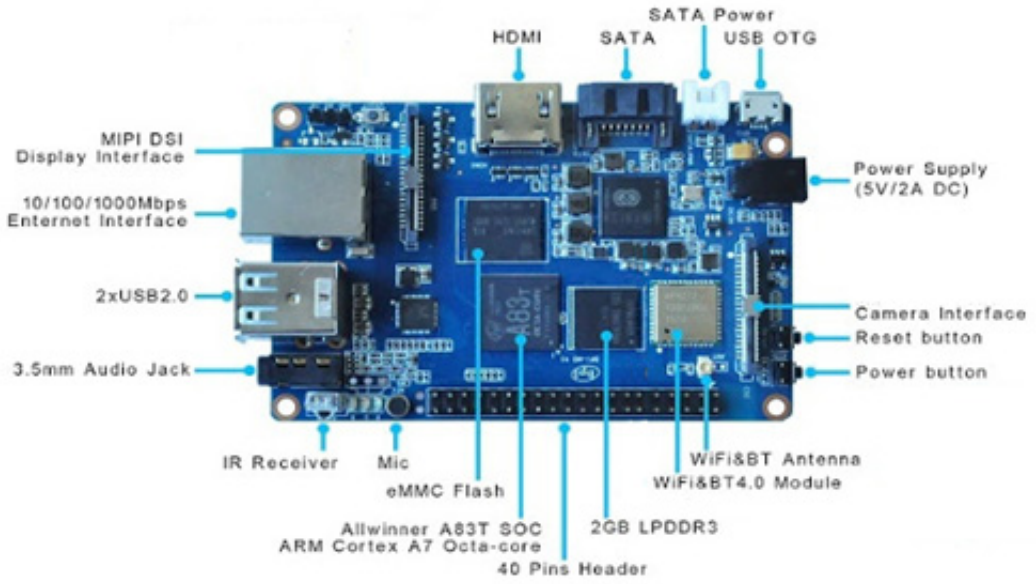
Banana Pi M3 é um dispositivo embarcado que possui oito núcleos de processamento, com processador fabricado pela AllWinner, modelo A83t e *Clock* de 2ghz, direcionado para processamento de aplicações móveis de alto desempenho. Possui ainda 2GB de memória RAM DDR3, o que o torna bastante eficiente para executar vários sistemas operacionais, como por exemplo, Android, Lubuntu, Ubuntu Mate, Debian e Raspbian. A figura 3 ilustra o dispositivo Banana Pi M3, assim como suas principais características.

Vale ressaltar que sua GPU PowerVR SGX544MP1 da Mediatek também executa com 8 *cores*, o que torna possível assistir vídeos em 4k sem travamentos. Além de diversos outros opcionais interessantes: *Wifi*, *Bluetooth* 4.0, entrada SATA, Gigabit Ethernet, eMMC de 8gb, microfone, receptor infravermelho, usb, otg e botão de *Reset* e *Power*.

2.3.1.3 Orange Pi Plus 2

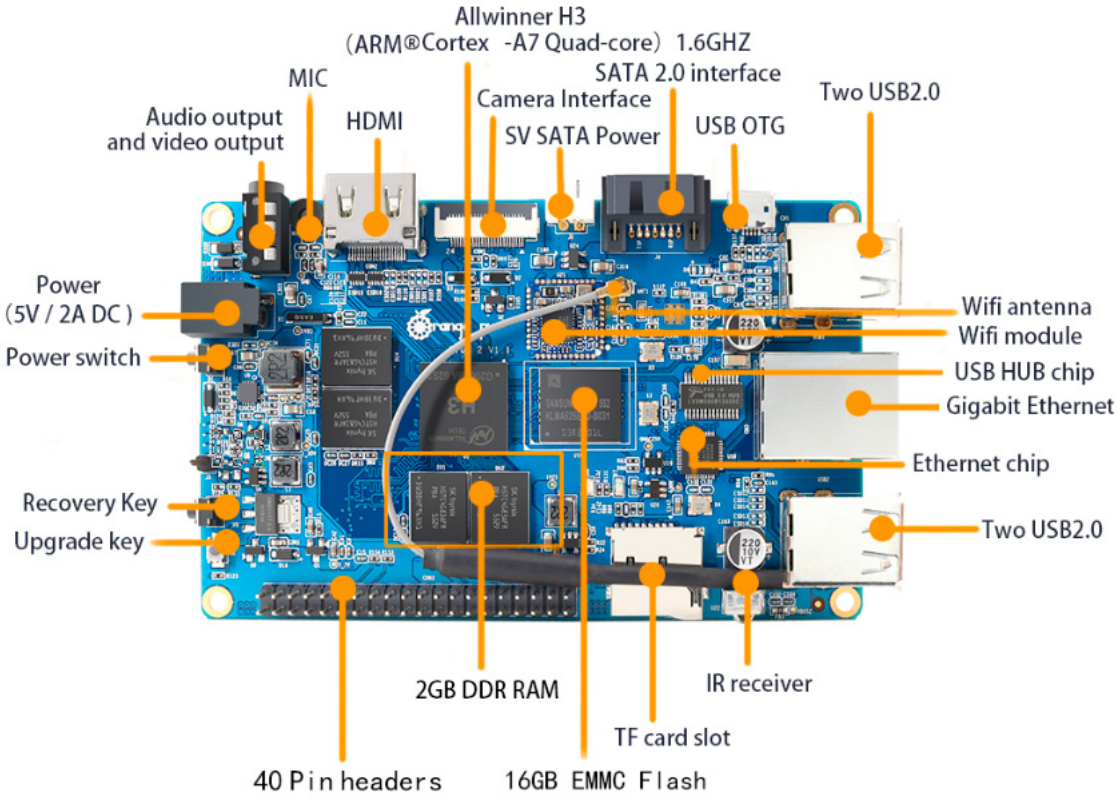
O dispositivo embarcado da Orange Pi Plus 2 é interessante quando suas configurações são comparadas a outras placas do mercado na mesma faixa de preço. O mesmo possui CPU com: 4 núcleos de processamento e *clock* de 1.6GHZ; aceleração gráfica por *hardware*; 2GB de memória RAM DDR3 compartilhada com a GPU; 4 portas USB Host e 1 micro-USB OTG; conectividade por Ethernet 10/100/1000M RJ45 e WI-FI IEEE 802.11 b/g/n; e memória Flash (eMMC) de 16GB. A figura 4 ilustra o *hardware* do dispositivo Orange Pi Plus 2, assim como algumas de suas características.

Figura 3 – Banana Pi M3.



Fonte: Adaptado de Banana Pi (2017).

Figura 4 – Orange Pi Plus 2.



Fonte: Adaptado de Orange Pi Plus 2 (2017).

O dispositivo embarcado Orange Pi Plus 2 possui diversas aplicações e funcionalidades. Como destaque é possível citar a sua utilização como computador portátil, servidor de aplicações

wireless, servidor de músicas e console de *Games*. No entanto, esse dispositivo embarcado pode ser utilizado como protótipo de diferentes projetos com inúmeras funcionalidades.

3

Trabalhos Correlatos

Neste capítulo são abordados todos os trabalhos classificados como significantes e correlatos à presente dissertação. Foi realizada uma análise sistemática com intuito de encontrar trabalhos relevantes em bases consideradas importantes na área da computação (*IEEE Xplore*, *Science Direct* e a Biblioteca Digital Brasileira de Computação). A quantidade de trabalhos que aborda o Asterisk em dispositivos embarcados é bastante reduzida. Dessa forma, também foram pesquisados trabalhos sobre análise de performance e eficiência energética com o uso do protocolo SIP e IAX2.

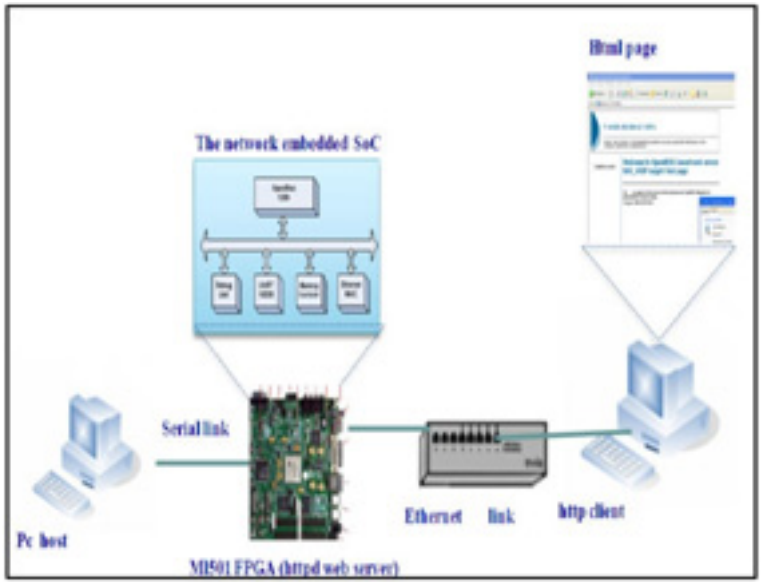
3.1 *Gateway* PBX-IP

Abid et al. (2012), em seu artigo "Embedded Implementation of an IP-PBX/VoIP Gateway", propõem a ideia de projetar e implementar um *gateway* PBX-IP embarcado, o qual utiliza soluções de baixo custo e de código aberto. O sistema integra *hardwar*e FPGA e incorpora um *software* na memória externa.

Seu experimento é composto por um *hardware* ML501 FPGA, com o *software* asterisk embarcado e dois computadores pcx86, sendo um conectado a porta serial e outro conectado a rede ethernet, conforme mostra a figura 5.

Os autores dividiram os testes em etapas. A primeira foi a configuração e compilação do sistema operacional embarcado, onde ocorreu a compilação do sistema operacional linux 3.0 na plataforma ML501 FPGA. A segunda foi a comunicação entre o sistema operacional e o *hardware*, implementado através dos *drivers* dos dispositivos. Já a terceira e última, foi o teste de aplicação, que consistiu no acesso via rede ethernet ao servidor web HTTP que se encontra no sistema embarcado.

Figura 5 – Teste *Gateway* PBX-IP.



Fonte: (ABID et al., 2012).

3.2 Análise de performance VoIP com Alix 2D2

Villacís, Acosta e Lara Cueva (2013), em seu artigo "Performance Analysis of VoIP Services over WiFi-based systems", apresentaram uma análise de desempenho do *hardware* Alix para utilização em sistemas VoIP sob redes *Wireless Fidelity* (Wifi), que adota um servidor com o *software* Asterisk embarcado. O experimento foi realizado com um *hardware* Alix 2D2, processador 500Mhz, 256mb de memória RAM, duas portas USB, dois slots miniPCI, uma porta serial, duas portas rj45 e um *slot* para cartão *compact flash* que funciona como disco rígido. Segundo os autores, o único sistema operacional suportado no *hardware* Alix, com suporte ao Asterisk, é o Voyage One 0.5.2 release, baseado em Linux e com Asterisk integrado.

Villacís, Acosta e Lara Cueva (2013), realizaram uma análise de desempenho dos protocolos SIP e IAX com os CODECS GSM, G.711 u/a, SPEEX e G.726 em redes Wi-fi. Esta análise consiste em testes de consumo da unidade central de processamento (CPU) e memória RAM. Para realizar o experimento, os autores alocaram um computador para executar a ferramenta SIPP, de modo a realizar um determinado número fixo de chamadas simultâneas; e um segundo computador com um *softphone* instalado, a fim de verificar a qualidade da chamada e determinar se a placa Alix suporta o número específico de chamadas. A ferramenta SIPP possibilita diminuir ou aumentar o número de chamadas, e assim permite chegar a um número apropriado de chamadas simultâneas que possa ser processado. Para monitorar a CPU e a memória RAM, os autores conectaram um cabo serial na placa Alix.

Os experimentos foram divididos em dois grupos. O primeiro grupo foi dividido em duas etapas: a primeira etapa baseada na implementação do protocolo SIP e a segunda baseada na implementação do protocolo IAX. Ambas sobre sistema Wi-Fi em conformidade com o padrão

IEEE 802.11b. Dessa forma, verificou-se a percentagem de CPU e uso de memória RAM. Foram realizados quatro testes: o primeiro com solicitação HTTP via *browser* a uma taxa de 1 repetição por segundo; o segundo realizou a medição de uma chamada entre o *softphone* X-pro com o protocolo SIP, de modo a utilizar o mesmo CODEC em ambas as extremidades; o terceiro teste utilizou diferentes CODECS; já o quarto e último teste, realizou uma chamada para o ramal 500 que fornece informações do próprio servidor, contudo, utilizou o mesmo *softphone*.

O segundo grupo abordou o padrão IEEE 802.11g, e realizou dois testes: um com protocolo SIP sem solicitação HTTP e outro com protocolo SIP com um pedido HTTP a uma taxa de 1 petição por segundo.

3.3 Avaliação de QoS com uso do SIP e IAX2

Edan et al. (2016), apresentaram em seu artigo uma avaliação de desempenho e Qualidade de serviço (Qos) de transmissão multimídia (voz e vídeo). Para isso, utilizaram os protocolos SIP e IAX2, baseado em um servidor Asterisk. A qualidade de serviço avaliada utilizou alguns parâmetros de Qos, como largura de banda, *jitter* e *delay*, a fim de investigar o desempenho de diferentes CODECS de voz e vídeo. Os seguintes CODECS foram avaliados: G.711 (ulaw e alaw), GSM, G.722, SPEEX, H263, H.264, H.261 e H.263P.

Para realizar o experimento, os autores utilizaram os *softwares* AsteriskNow PBX 6.12, os *softphones* X-lite e Zoiper, e o analisador de protocolo Wireshark. Vários *hardwares* foram utilizados, como exemplo temos: dois computadores core i7 com 8g de memória RAM utilizados como servidores, um *access point* NetCommWireless, um *switch* e dois *Iphones* conectados a rede local por Wifi. Adotou-se a estratégia de realizar quatro tipos de chamadas, sendo SIP-SIP, IAX2-IAX2, SIP-IAX2 e IAX2-SIP. Assim, utilizou separadamente um e dois servidores interligados.

Com o uso da rede ethernet, os resultados demostraram que as chamadas de voz IAX2-IAX2 e IAX2-SIP obtiveram os melhores resultados na qualidade da chamada, na menor largura de banda e *delay*. No entanto, com a chamada IAX2-SIP, houve um aumento no consumo de banda.

Ao utilizar os mesmos passos com a rede Wi-fi, encontraram um resultado semelhante de largura de banda em relação aos protocolos IAX2-IAX2 e IAX2-SIP. Porém, o *jitter* foi maior. Os testes de chamadas de voz com dois servidores interligados com o uso do protocolo SIP também obtiveram resultados semelhantes. A interligação de servidores com IAX2 via ethernet demonstrou utilizar menor largura de banda. Já com o Wi-fi, a largura de banda foi inferior, mas o *jitter* e o *delay* foram maiores.

Diante dos testes realizados, os autores concluíram que ao utilizar o protocolo IAX2 para estabelecer uma chamada de voz entre duas extremidades, com o uso ou não de dois servidores

Asterisk interligados, eles têm melhores resultados comparado ao protocolo SIP. No entanto, o *jitter* com o IAX2 é maior. Além disso, as chamadas de vídeo com o uso do protocolo IAX2 possuem qualidade de vídeo inaceitável. Os mesmos testes foram realizados com o protocolo SIP e obtiveram um desempenho excelente. Os melhores resultados ocorreram na rede cabeada, com destaque para o CODEC de voz G.711a/u, o qual demonstrou melhor desempenho na qualidade da chamada.

3.4 Implementação e avaliação de um Sistema de Comunicação Unificada

Tesfamicael et al. (2014), em seu artigo "Implementation and Evaluation of Open Source Unified Communications for SMBs" apresentaram a implementação e avaliação de um sistema de comunicação unificada, composto por mensagens instantâneas e compartilhamento, conferências (voz e vídeo), mensagens de voz, comunicação VoIP e mobilidade. Para realizar o experimento, os autores utilizaram dois servidores, um virtualizado em uma plataforma Vmware e outro físico. Utilizou-se também um provedor de Internet a fim de prover a conexão dos servidores com a rede pública e uma conexão com a operadora de telecomunicações. O protocolo utilizado foi o SIP, e as configurações de chamadas foram realizadas no arquivo sip.conf. O FreePBX foi o sistema de comunicação abordado, um sistema *open source* baseado em Asterisk.

O experimento se deu com os dois servidores Asterisk interligados através de um roteador cisco 2811. Realizou-se também a configuração do SIPP, e um cliente SIP para realizar testes com suporte para várias chamadas simultâneas. Para a comunicação por mensagens instantâneas utilizou-se o Openfire. O objetivo da pesquisa foi simular em um ambiente real as limitações do servidor asterisk através de testes de desempenho.

Os autores observaram que o CODEC G.729 consome menos CPU que o CODEC G.711. O servidor Asterisk Core i7-3770 CPU @ 3.40 Ghz, utilizado no experimento, suporta aproximadamente 300 chamadas simultâneas com o uso do CODEC G.711. Ao empregar a transcodificação (troca de CODECS), ele suportou aproximadamente 200 chamadas simultâneas. No entanto, para locais com largura de banda limitada, recomenda-se o uso do CODEC G.729.

3.5 Considerações sobre os Trabalhos Correlatos

Os trabalhos apresentados são de conteúdo relevante, abordam uma prática investigativa para o conceito de avaliação de desempenho de um sistema de comunicação VoIP, que utiliza os protocolos SIP e IAX. No entanto, os autores Tesfamicael et al. (2014) iniciaram seu trabalho com uma abordagem de um sistema de comunicação unificada. Porém, em seus experimentos realizaram apenas testes de comunicação por voz e vídeo, assim não demonstram a eficiência do sistema quando utilizado com vários módulos do sistema proposto de comunicação unificada.

Ainda em Tesfamicael et al. (2014), os autores realizaram um trabalho com grande contribuição para a indústria, pesquisadores e comunidade acadêmica. Porém, ainda há muito o que ser feito, como por exemplo, a realização desses mesmos experimentos em dispositivos de *hardware* embarcados, bem como a medição da eficiência energética em momentos de elevado consumo do sistema, não abordados nessa pesquisa.

No trabalho de Villacís, Acosta e Lara Cueva (2013), o melhor desempenho produziu-se com o uso do protocolo IAX e o CODEC GSM, o qual obteve sucesso em 110 chamadas telefônicas simultâneas. Trata-se de uma pesquisa com conteúdo significativo. No entanto, o *hardware* abordado foi descontinuado pelo fabricante. Os autores poderiam ter realizado uma análise de eficiência energética no momento de elevado número de chamadas simultâneas, bem como ter feito testes do comportamento de chamadas simultâneas com transcodificação. Dessa forma, teriam realizado uma pesquisa mais criteriosa.

Já no trabalho de Edan et al. (2016), os autores poderiam realizar os testes do comportamento de chamadas simultâneas, tanto com os *softphones* quanto entre os servidores. Poderiam também projetar e desenvolver um aplicativo para transmissão de vídeo com suporte ao protocolo IAX2, de modo a possibilitar a conclusão de um dos seus testes elencados nesse trabalho. Na tabela 2 constam os *hardwares*, *softwares*, protocolos e CODECS utilizados nos experimentos dos trabalhos correlatos.

Abid et al. (2012) realizaram meramente um teste de suporte ao *software* Asterisk no *hardware* embarcado FPGA. Os autores realizaram apenas um teste de conectividade em rede. Porém, não realizaram testes para aferir o quantitativo de chamadas simultâneas suportadas, medição do consumo de energia, processamento e memória.

Nota-se o uso de várias versões do sistema operacional Linux, bem como o uso do *software* Asterisk em dispositivo embarcado e em *hardwares* distintos. Além disso, todos os trabalhos realizaram experimentos em ambiente real, o que deixa a pesquisa mais produtiva. Os autores Tesfamicael et al. (2014) e Villacís, Acosta e Lara Cueva (2013) abordaram o uso da ferramenta SIPP, a qual possibilita realizar várias chamadas simultâneas. O protocolo SIP, predominante em três trabalhos, foi utilizado com maior frequência, bem como o CODEC G.711. Já o trabalho de Abid et al. (2012) abordou apenas o suporte ao *hardware* FPGA com o sistema Asterisk.

Um bom trabalho de pesquisa necessita de parâmetros para validação de resultados. Entretanto, os autores não abordaram nenhuma metodologia para validação, o que conduz a acreditar em ocorrência de possíveis erros nos resultados demonstrados.

A tabela 3 demonstra uma comparação entre este trabalho e os demais. Observa-se que este é mais completo. Nos capítulos seguintes serão descritos os detalhes da implementação da análise de desempenho e da eficiência energética em dispositivos embarcados com uso do Asterisk, bem como a prototipação e medições realizadas para extração dos resultados.

Tabela 2 – Comparação entre os trabalhos correlatos.

Autores	Hardware	Software	Protocolo	CODEC	Experimento	Simulado	Real
Edan et al(2016)	PC X86 i7 8gb RAM AcessPoint Switch Iphone	AsteriskNow X-lite Zoiper Wireshark	sip,iax2	G.711u/a Gsm G.722 Speex H.263 H.264 H.261 H.263p	x		x
Tesfamicael et al (2014)	PC x86 i7 3.40Ghz 8gb RAM	CentOS Ubuntu FreePBX Sipp Openfire	sip	G.711u/a G.729 Gsm	x		x
Villacis et al (2013)	Alix 2d2	Voyage Asterisk Sipp	sip,iax	G.711u/a G.722 Gsm Speex G.726	x		x
Abid et al (2012)	ML501 FPGA	Linux Asterisk	-	-	x		x
Esta dissertação	Raspberry Pi 3 Banana Pi M3 Orange Pi Plus 2 Mikrotik 951g	Raspbian Armbian Wireshark Zabbix Zoiper X-Lite	sip, iax	G.711u G.711a G.722 Gsm Ilbc Speex	x		x

Fonte: Autoria própria.

Tabela 3 – Comparação de pesquisas dos trabalhos correlatos.

Qualidades	Edan et al. (2016)	Tesfamicael et al. (2014)	Villacis et al. (2013)	Abid et al. (2012)	Esta dissertação
Dispositivo embarcado	-	-	x	x	x
Vazão, *jitter*, *delay*	x	-	-	-	x
Chamadas simultâneas sip	-	x	x	-	x
Chamadas simultâneas iax2	-	-	x	-	x
Consumo de memória	-	-	x	-	x
Consumo de CPU	-	x	x	-	x
Consumo de energia	-	-	-	-	x
Metodologia p\validação	-	-	-	-	x

Fonte: Autoria própria.

4

Metodologia para Prototipação e Coleta de Dados

Nesta seção são apresentados métodos utilizados para a prototipação e coleta dos dados de consumo nos dispositivos embarcados implementados, tais como: metodologia utilizada, arquitetura de simulação ou cenário de testes, dispositivos embarcados Raspberry Pi 3, Banana Pi M3 e Orange Pi Plus 2, protótipo para medição do consumo de energia. Assim como as ferramentas Wireshark e Zabbix, utilizadas para coletar e analisar os dados de desempenho das implementações, as quais são detalhadas neste capítulo.

4.1 Metodologia

Ao definir uma metodologia de avaliação de desempenho, deve-se ter atenção a fim de não cometer erros comuns, como inexistência de objetivos, propostas tendenciosas, incorretos métodos de avaliação, entre outros. Para evitar tais erros, o ideal é adotar uma abordagem sistemática como a proposta por Jain (1991), a qual foi aplicada neste trabalho. Para empregar essa metodologia torna-se necessário seguir uma sequência de passos.

O primeiro passo é a definição dos objetivos e do sistema. Pode ser difícil precisar os objetivos, todavia, é o primeiro passo para a resposta de um problema. O sistema deve ser claro, pois ele influencia diretamente nas métricas utilizadas para avaliação de desempenho, bem como no fluxo de carga para validação.

O segundo passo é a elaboração da lista de serviços e resultados esperados. Cada sistema provê um conjunto de serviços, e para cada um deles existe um conjunto de possíveis resultados.

Como terceiro passo, tem-se a seleção de métricas que estabelecem os critérios para a comparação do desempenho. A fim de escolher boas métricas, Jain (1991) recomenda verificar se as mesmas possuem baixa variação para diminuir o número de experimentos necessários; não redundância de modo a evitar números supérfluos; e completude com o intuito de retratar todos os resultados possíveis. Ainda segundo Jain (1991), as métricas mais comuns são: eficiência (ca-

pacidade utilizável sobre capacidade nominal), confiabilidade (tempo sem erros), disponibilidade (tempo médio entre falhas) e a relação custo/desempenho.

O quarto passo é a elaboração da lista de parâmetros que afetam o desempenho. Essa lista pode ser dividida em parâmetros de sistema (que geralmente não variam de uma instância para outra) e de carga (que são características das solicitações dos usuários).

Já o quinto passo, trata da escolha de fatores para estudo, que são parâmetros os quais sofrerão variações durante a pesquisa. Esses fatores podem assumir valores que são denominados níveis. Recomenda-se iniciar com poucos fatores e poucos níveis, e aumentar a lista conforme necessidade.

O sexto passo é a seleção da técnica de avaliação. Existem três técnicas: a simulação, a modelagem analítica e a medição. A técnica a utilizar dependerá do tempo e dos recursos disponíveis para a resolução do problema.

Como sétimo passo, tem-se a escolha da carga, a qual consiste em uma lista de solicitações de serviços ao sistema. É importante que retrate o uso real.

Em seguida, o oitavo passo é o planejamento dos experimentos. Com a lista de fatores e níveis, deve-se estabelecer uma sequência de experimentos de modo a obter o máximo de informações possíveis com o mínimo de esforço realizado.

Como nono passo, tem-se a análise e interpretação dos dados. Nessa etapa deve-se utilizar técnicas estatísticas adequadas a fim de consolidar os resultados obtidos, de modo a permitir realizar conclusões sobre o desempenho do sistema.

Já o décimo e último passo é a apresentação dos resultados. Nesta etapa deve-se atentar à apresentação final da avaliação.

4.1.1 Aplicação da Metodologia

Ao aplicar esta metodologia, nota-se sua importância diante da forma organizada na qual o trabalho foi conduzido. Inicialmente, torna-se necessário definir os objetivos, em seguida o escopo do sistema, os serviços oferecidos, e a técnica de avaliação. Como são mostrados a seguir.

Objetivos:

- Realizar uma análise de desempenho e eficiência energética em três dispositivos embarcados de baixo custo;
- Determinar fatores relevantes no que se refere ao desempenho destes equipamentos.

Sistema:

- O sistema corresponde a um *software* de comunicação por voz sobre IP denominado Asterisk, o mesmo interage com o meio através do recebimento e realização de chamadas telefônicas através do Protocolo IP.

  Serviço:

- Comunicação de voz sobre IP.

  Técnica de avaliação:

- Medição, pois trata-se de uma técnica útil para análise de desempenho de sistemas informáticos.

  A atividade foi dividida em cinco etapas:

- Etapa 1 - Projeto e cenário de testes;
- Etapa 2 - Especificação das métricas;
- Etapa 3 - Definição dos parâmetros, fatores e carga;
- Etapa 4 - Planejamento e realização dos experimentos;
- Etapa 5 - Análise estatística dos resultados obtidos.

A subseção 4.1.2 descreve as três primeiras etapas, enquanto a quarta e quinta são abordadas no próximo capítulo.

4.1.2 Projeto e Cenário de Testes

Para realizar a prototipação deve-se pressupor a existência de um modelo computacional idêntico ao ambiente real de produção. Na literatura, boas descrições para análise de desempenho em sistemas de comunicação por voz sobre IP foram encontradas, como exemplo é possível citar os trabalhos de Villacís, Acosta e Lara Cueva (2013) e Edan et al. (2016). O primeiro trabalho faz uma análise de desempenho do *hardware* Alix 2D2 para utilização em sistemas VoIP. Já o segundo apresenta uma avaliação de desempenho e Qualidade de serviço (Qos) de transmissão multimídia (voz e vídeo).

Baseado nesses trabalhos e em especificações técnicas de equipamentos disponíveis comercialmente, o presente trabalho foi realizado com três dispositivos embarcados modernos e com a proposta de complementar as pesquisas já realizadas, e assim contribuir de forma efetiva para a indústria das telecomunicações, para pequenas e grandes empresas. Bem como para a área acadêmica, pois amplia a realização de novas pesquisas.

Este trabalho verifica a possibilidade de atestar a utilização de dispositivos embarcados modernos como servidores de um sistema de comunicação por voz sobre IP denominado Asterisk, com o uso dos protocolos SIP e IAX2 de modo a utilizar os seguintes CODECS: G.711a, G.711u, Gsm, Speex, Ilbc e G.722. Não foram utilizados outros CODECS devido as limitações encontradas no trabalho realizado, tais como: *softphones* gratuitos utilizados só possuírem suporte a esses CODECS; e possíveis erros na instalação dos CODECS nos dispositivos embarcados. Os dispositivos embarcados abordados neste trabalho possuem preço médio de $50 dólares. Assim, é possível considerar como uma solução de baixo custo.

A análise de desempenho consiste em verificar a vazão, *jitter* e *delay* das chamadas em cada dispositivo embarcado com os protocolos e CODEC'S anteriormente citados. Para isso, utilizou-se a ferramenta para análise de tráfego Wireshark, *software* especializado em análise de tráfego em redes IP, bastante utilizado por pesquisadores no meio acadêmico, como por exemplo, (EDAN et al., 2016).

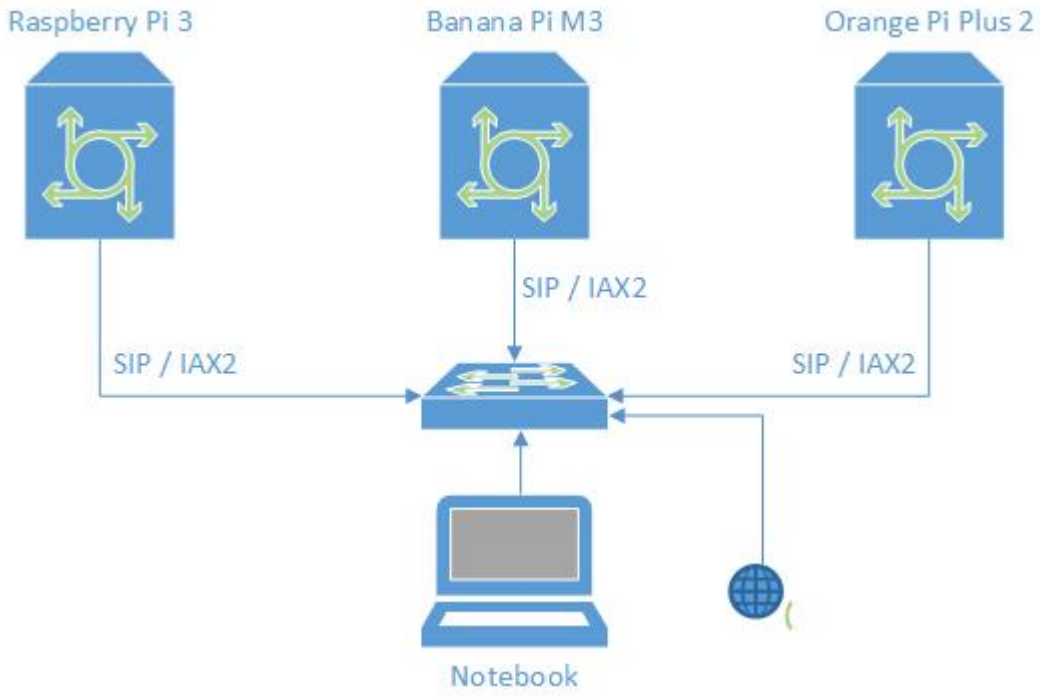
Ainda em análise de desempenho, realizou-se o monitoramento do consumo de memória e processamento em momentos de elevado uso do sistema; a fim de atingir o objetivo desejado utilizou-se o *software* de monitoramento Zabbix. Já a medição da eficiência energética foi realizada através de um protótipo de medição de corrente elétrica utilizado por Maia (2017), o qual permitiu monitorar a eficiência energética durante todo o experimento.

Para a realização deste trabalho, foi necessário elaborar um cenário de testes, de modo a possibilitar a execução de todos os objetivos propostos. Sendo assim, o cenário contemplou três dispositivos embarcados modernos e de baixo custo (Raspberry Pi 3, Banana Pi M3 e Orange Pi Plus 2), um RouterBoard 951g Mikrotik com 5 portas 10/100/1000 Gigabit Ethernet, que realizou o *switch*, um Notebook Core i7 8GB RAM e 500GB de HD, e um aparelho telefônico IP Yalink T19p. A figura 6 ilustra a arquitetura do cenário para a realização dos experimentos.

O RouterBoard Mikrotik, que realiza o *switch*, é responsável por interligar todos os equipamentos em rede. Cada dispositivo embarcado possui suporte ao sistema de comunicação por voz sobre IP Asterisk. O *notebook* foi alocado para ser o gerador de carga e realizar a coleta dos dados. Já o telefone IP possui o intuito de aferir a qualidade da chamada em momentos de elevado consumo de carga ou fluxo de chamadas.

Os dispositivos foram submetidos a momentos de elevado consumo do sistema. Para isso, uma máquina virtual foi instalada no *notebook*, a qual possibilitou realizar diversas chamadas simultâneas, de modo a atingir o nível máximo de chamadas suportadas em cada dispositivo embarcado. Os testes foram realizados com os Protocolos SIP e IAX2, bem como com os CODEC'S anteriormente mencionados. Assim, foi possível atingir o objetivo deste trabalho com a realização das devidas medições.

Figura 6 – Projeto do cenário de testes.



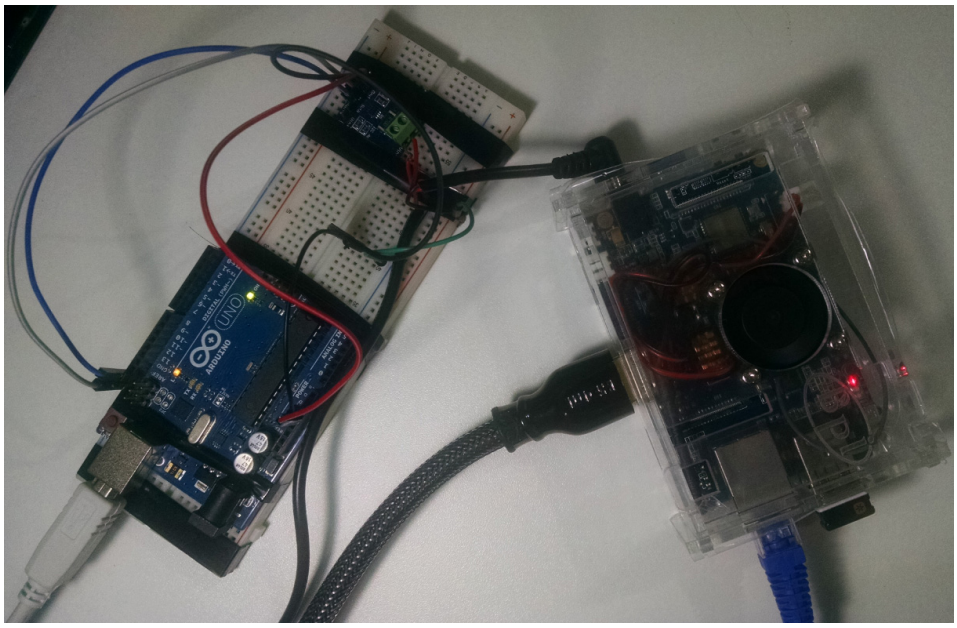
Fonte: Autoria própria.

4.2 Sensor de Corrente

Com o objetivo de realizar uma análise de eficiência energética através da medição de corrente e tensão elétrica nos dispositivos embarcados em momentos de elevado consumo de memória e processamento, utilizou-se um protótipo abordado por Maia (2017), o qual realiza uma medição física. O mesmo é constituído por um dispositivo embarcado Arduino Uno, que realiza a comunicação com a placa de monitoramento de tensão e corrente Adafruit. Essa, por outro lado, utiliza um sensor de corrente e tensão INA219, desenvolvido pela empresa Texas Instruments.

A figura 7 mostra o protótipo de medição durante uma medição. A coleta dos dados se deu via porta serial no dispositivo Arduino, o qual é conectado a um microcomputador, onde ocorre a programação e exibição dos dados através de *software* próprio do dispositivo.

Figura 7 – Sensor de corrente.



Fonte: Autoria própria.

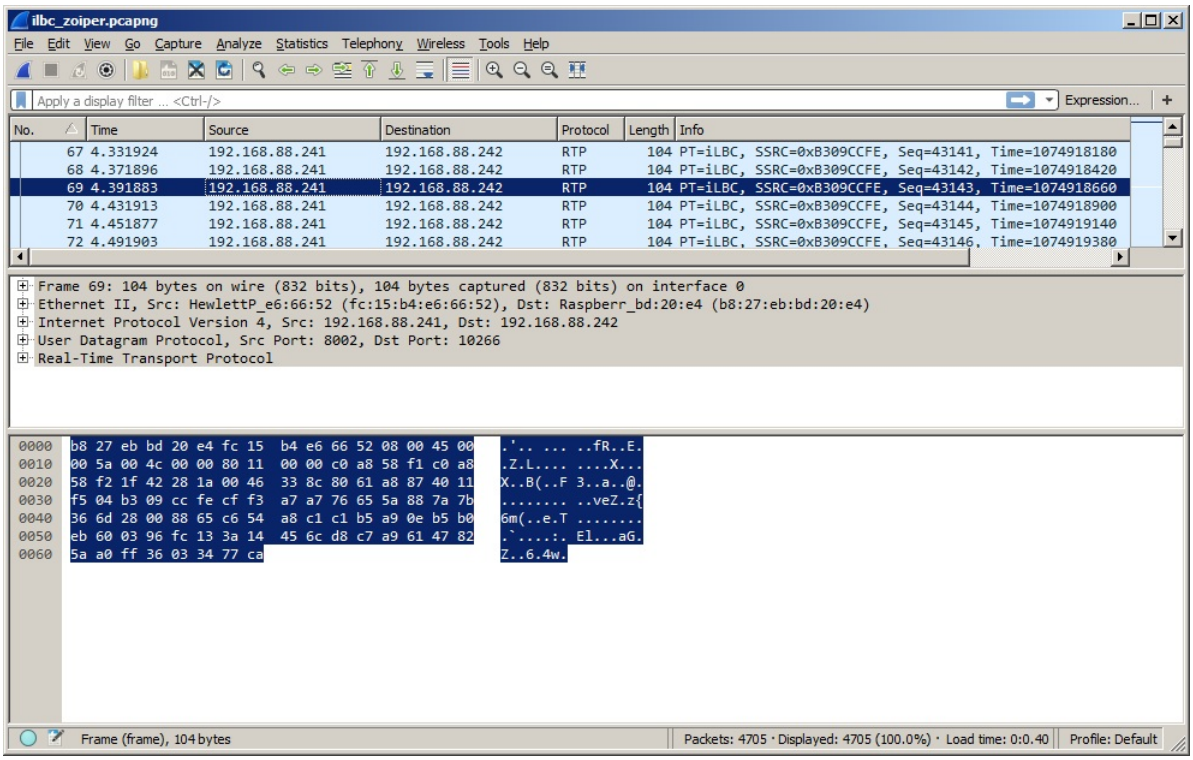
4.3 Wireshark

Para Shimonski (2014), trata-se de um *software* que realiza análise de protocolos em redes de computadores cuja finalidade consiste em detectar e solucionar possíveis problemas; bastante utilizada por administradores de redes. Analisadores de rede permitem capturar todo o tráfego que passa pela rede, decifrar e interpretar inúmeros protocolos utilizados.

Ainda de acordo com Shimonski (2014), os dados decifrados são interpretados e demonstrados, de modo a facilitar a sua compreensão, por meio da remoção de camadas dos dados encapsulados utilizados para identificá-los ou para permitir que possam ser usados na rede. Um analisador de rede também pode capturar somente o tráfego que corresponda a critérios de seleção definidos por um filtro. Isso permite que um profissional capture somente o tráfego que seja relevante ao problema em questão.

A figura 8 demonstra a ferramenta Wireshark com dados capturados e prontos para inspeção.

Figura 8 – Wireshark.

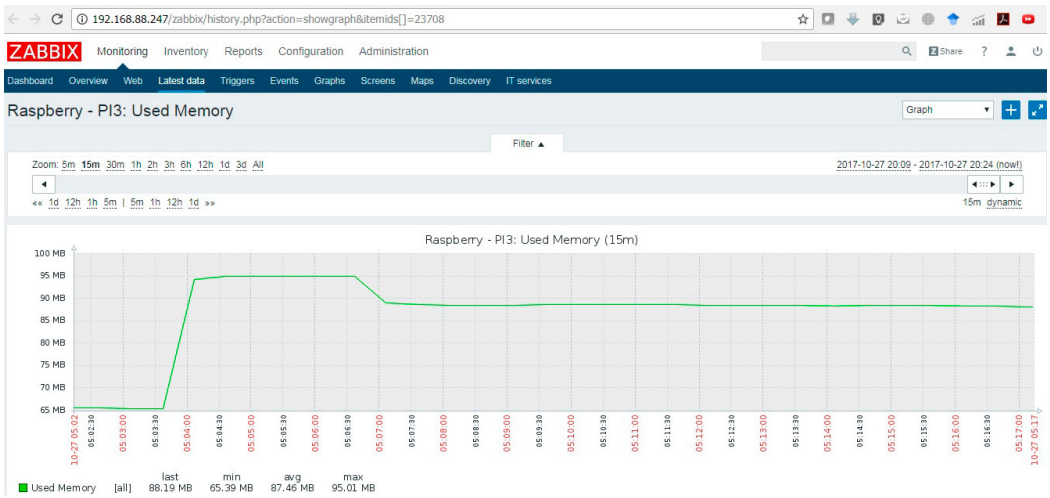


Fonte: Autoria própria.

4.4 Zabbix

Zabbix é um *software* consolidado como ferramenta de monitoramento em redes de computadores, servidores e serviços. O mesmo possui o intuito de monitorar a integridade, disponibilidade, experiência de usuário e qualidade de serviços. A figura 9 demonstra um monitoramento de memória de um dispositivo embarcado realizado com o *software* Zabbix.

Figura 9 – Zabbix.



Fonte: Autoria própria.

Segundo Dalle Vacche e Kewan Lee (2015), o Zabbix surgiu em 2001 e desde o seu lançamento se distinguiu como uma solução de monitoramento poderosa e eficaz. Trata-se de um produto de código aberto, fácil de obter e implantar. Sua abordagem possui métricas e alarmes que ajudam bastante o administrador de redes. Fornece ótimos relatórios, assim como a possibilidade de visualizar dados de recursos com base nas informações armazenadas. Possui ainda um excelente mecanismo de notificação, o qual possibilita aos usuários configurar alertas para qualquer evento.

O Zabbix é uma solução ideal para pequenos e grandes ambientes distribuídos, onde é possível gerenciar de forma eficiente e extrair informações significativas de objetos e eventos.

5

Experimentos e Resultados

Este capítulo apresenta a implementação do experimento, que consiste em: realizar a montagem do cenário de teste com os dispositivos embarcados de forma individual; efetuar a abordagem dos *softwares* necessários nos dispositivos para a elaboração do experimento; proceder com o processo de coleta do *jitter*, vazão e *delay* com o *software* Wireshark; implementar um código em *Shell Script* com a finalidade de gerar cargas de chamadas SIP e IAX com os CODEC's G.711u (Ulaw), G.711a (Alaw), Gsm, Speex, Ilbc e G.722; executar elevados fluxos de chamadas e em paralelo realizar o monitoramento do consumo de processamento, memória e energia; e por fim, aferir os resultados.

5.1 Elaboração do Cenário de Testes

Ao realizar a montagem do cenário para elaboração dos testes, foi necessário instalar um sistema operacional nos dispositivos embarcados. Para isso utilizou-se o Armbian no Banana Pi M3 e Orange Pi Plus 2. Já no Raspberry Pi 3, foi instalado o Raspbian, pois não foi encontrado Armbian para o dispositivo Raspberry. Ambos os sistemas operacionais são baseados no GNU Linux Debian 8. Em sequência, ocorreu a instalação do *software* de comunicação por voz sobre IP Asterisk.

Após a instalação dos sistemas operacionais e sistema de comunicação por voz sobre IP, foi preciso instalar dissipadores de calor, assim como um *cooler*, a fim de refrigerar os dispositivos, pois houve um elevado número de ocorrências de mensagens de alarme referindo-se a alta temperatura nos dispositivos.

Ocorreu também a instalação de duas máquinas virtuais no *notebook*. Em uma foi instalado o *software* de comunicação por voz sobre IP Asterisk e na outra o *software* de monitoramento Zabbix. Para a instalação dessas máquinas virtuais, utilizou-se o Oracle VM VirtualBox. No *notebook*, ocorreu também a instalação do *software* de captura de pacotes Wireshark, assim como

a instalação de dois *softphones* gratuitos: Zoiper e X-lite. O primeiro dá suporte aos CODEC's G.711a, G.711u, Gsm, Speex e Ilbc. Já o segundo acrescenta o G.722. A tabela 4 ilustra os *softwares* utilizados no experimento.

Tabela 4 – *Softwares* utilizados no experimento.

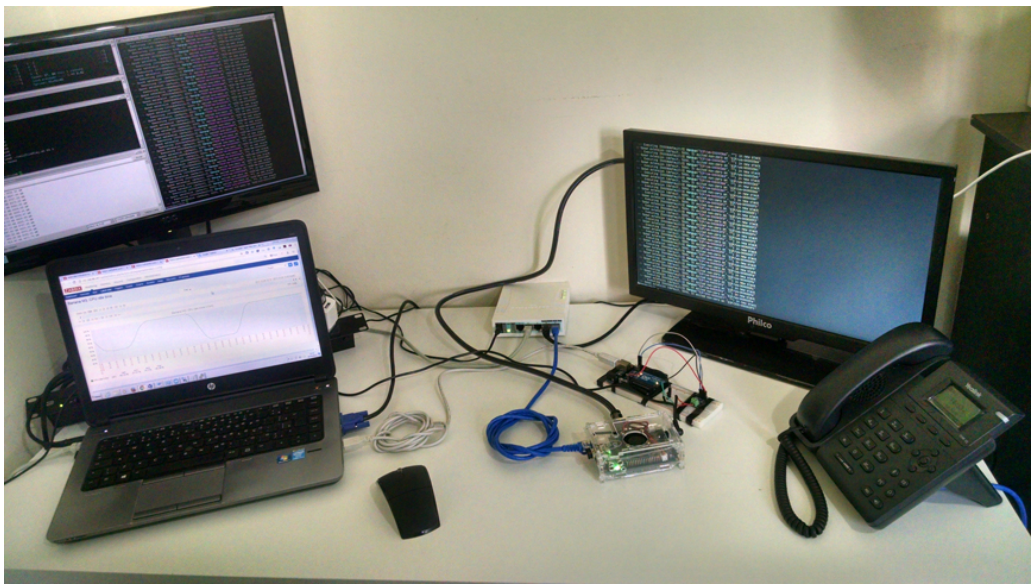
Dispositivos	***Notebook***	**VM VirtualBox**
ARMbian 5.25 Raspbian Asterisk 13	Wireshark Zoiper X-Lite VM Virtual Box	Asterisk 13 Zabbix

Fonte: Autoria Própria

Para validar o cenário de testes, dois ramais foram configurados em cada Asterisk nos dispositivos embarcados. Os *softphones* também foram configurados com a finalidade de executar as chamadas. Dessa forma, realizou-se algumas chamadas entre os ramais. Durante as chamadas foi realizado a captura de pacotes com o *software* Wireshark, bem como o monitoramento do processamento e memória com o *software* Zabbix, e o monitoramento da energia com o circuito INA219.

A intenção é realizar uma análise de desempenho e eficiência energética em 3 dispositivos embarcados com uso do *software* de comunicação por voz sobre IP Asterisk, a qual consiste em: realizar a coleta do *jitter*, vazão e *delay* com o *software* Wireshark; apurar o quantitativo de chamadas simultâneas suportadas em cada dispositivo; e em paralelo realizar medições do consumo do processamento, memória e energia. A figura 10 ilustra o cenário real em que os testes foram realizados.

Figura 10 – Cenário real de testes.



Fonte: Autoria própria.

5.2 Vazão, *Jitter* e *Delay*

Após a realização dos testes preliminares para validar o cenário que objetiva simular um ambiente real de produção, iniciou-se a coleta de vazão, *jitter* e *delay* das chamadas realizadas em cada dispositivo embarcado. De acordo com Voip-info (2017), o termo vazão compreende uma medida da capacidade de transmissão em uma rede, o *delay* é o atraso na entrega dos pacotes e o *jitter* é a variação do atraso entre os datagramas IP.

Também foi necessário realizar configurações no Asterisk. Para esse fim, configurou-se um ramal de número 299 para deixar em modo automático a chamada em espera e tocar uma música em seguida. Essa configuração foi realizada no arquivo extensions.conf através da função MusicOnHold. Dessa forma, ocorre o processamento de mídia e dispensa um segundo agente atendedor. A figura 11 ilustra as configurações do arquivo extensions.conf.

Figura 11 – Arquivo Extensions.conf.

```
1  [default]
2  exten => _2XX,1,Dial(SIP/${EXTEN})
3  exten => _2XX,2,Hangup()
4  exten => _1XX,1,Dial(IAX2/${EXTEN})
5  exten => _1XX,2,Hangup()
6
7  exten => _299,1,Answer
8  exten => _299,2,Set(CHANNEL(musicclass=default)
9  exten => _299,3,MusicOnHold()
10 exten => _299,4,Hangup
11
```

Fonte: Autoria própria.

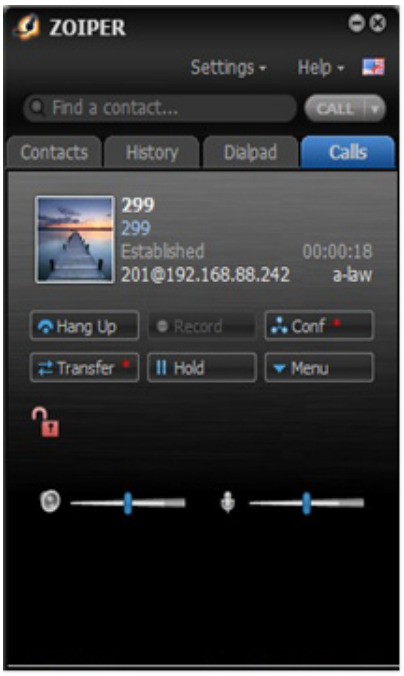
Chamadas foram realizadas entre os *softphones* e o ramal 299. Dessa forma, foi possível realizar a captura dos pacotes e consequentemente efetuar a análise. A coleta foi realizada com o protocolo SIP e IAX2, e com os CODEC's suportados pelos *softphones* gratuitos abordados, os quais foram listados na seção anterior. Todavia, totalizaram-se 33 coletas (6 SIP e 5 IAX2 em cada dispositivo embarcado), cada uma de chamadas com duração de aproximadamente 2 minutos. Assim, foi possível obter um quantitativo de pacotes para realizar a análise; aproximadamente 5.000 datagramas IP. A figura 12 ilustra uma chamada realizada para o ramal 299 com o *softphone* Zoiper. Já a figura 13 ilustra uma coleta realizada com o protocolo SIP e CODEC G.711a no dispositivo Raspberry Pi 3.

A ferramenta Wireshark possui filtros que realizam de modo automático a análise do *jitter*, vazão e *delay*. Contudo, utilizou-se a média, desvio padrão e intervalo de confiança de 95% como parâmetro para as medições. No entanto, os cálculos foram realizados manualmente, através da exportação dos dados capturados para o LibreOffice Calc.

As subseções a seguir apresentam os resultados coletados com os protocolos SIP e IAX2 nos dispositivos embarcados Raspberry Pi 3, Banana Pi M3 e Orange Pi Plus 2, com o seu

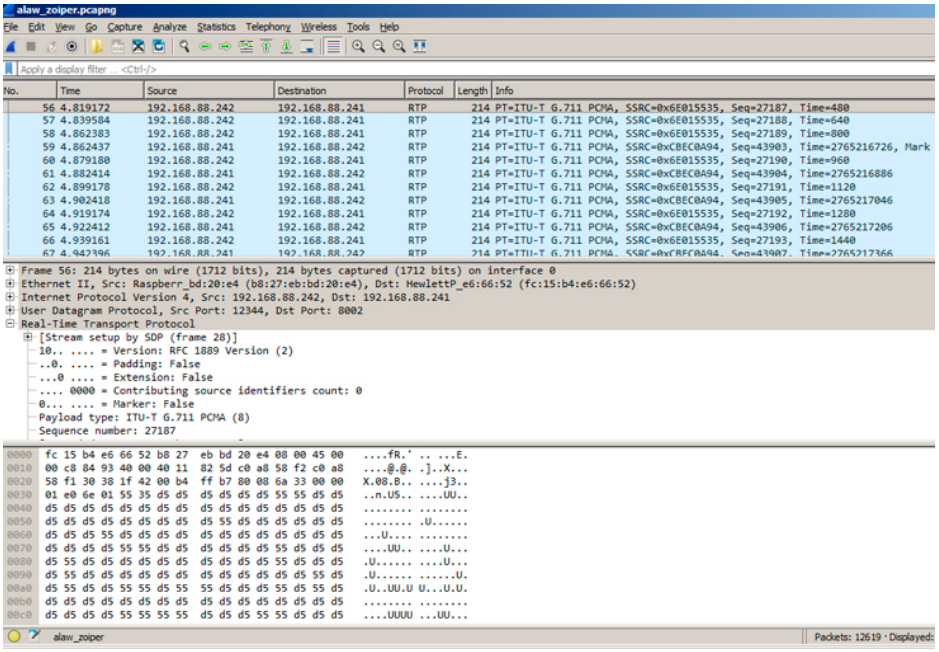
respectivo desvio padrão e intervalo de confiança de 95%, assim como breves discussões. O resultado da vazão encontra-se em kbps, já o *jitter* e *delay* em milissegundos. Não foi possível realizar medição do CODEC G.722 com o protocolo IAX2, devido ao limite de licença gratuita dos *softphones* abordados.

Figura 12 – Chamada para 299.



Fonte: Autoria própria.

Figura 13 – Coleta SIP com CODEC G.711a.



Fonte: Autoria própria.

5.2.1 Vazão

As tabelas 5 e 6 ilustram a vazão dos pacotes coletados, o resultado encontra-se em kbps. Nota-se que os CODEC'S G.711a, G.711u e G.722 possuem maior vazão e valores idênticos. Já os CODEC'S Gsm, Speex e Ilbc destacam-se por apresentar menor banda, tanto com o protocolo SIP quanto com o IAX2. Observa-se também, que os resultados encontrados nos três dispositivos embarcados são similares.

Tabela 5 – Coleta Vazão SIP. Unidade quilobits por segundo (Kbps).

Raspberry Pi 3 - Protocol SIP				
CODEC's	**Média**	**Desvio Padrão**	**Limite Inferior**	**Limite Superior**
G.711a	81,13	0,76	81,08	81,18
G.711u	81,11	0,77	81,06	81,15
Gsm	29,57	0,28	29,56	29,58
Speex	24,52	0,20	24,51	24,52
Ilbc	24,36	0,38	24,35	24,38
G.722	81,09	0,75	81,06	81,11
Banana Pi M3 - Protocol SIP				
G.711a	80,50	0,73	80,48	80,53
G.711u	80,48	0,72	80,46	80,5
Gsm	29,38	0,27	29,37	29,39
Speex	24,58	0,24	24,58	24,59
Ilbc	24,49	0,00	24,49	24,49
G.722	80,94	0,79	80,92	80,97
Orange Pi Plus 2 - Protocol SIP				
G.711a	80,51	0,74	80,49	80,53
G.711u	80,52	0,74	80,50	80,54
Gsm	29,40	0,27	29,39	29,41
Speex	24,58	0,23	24,57	24,58
Ilbc	24,49	0,00	24,49	24,49
G.722	81,06	0,76	81,04	81,08

Fonte: Autoria própria.

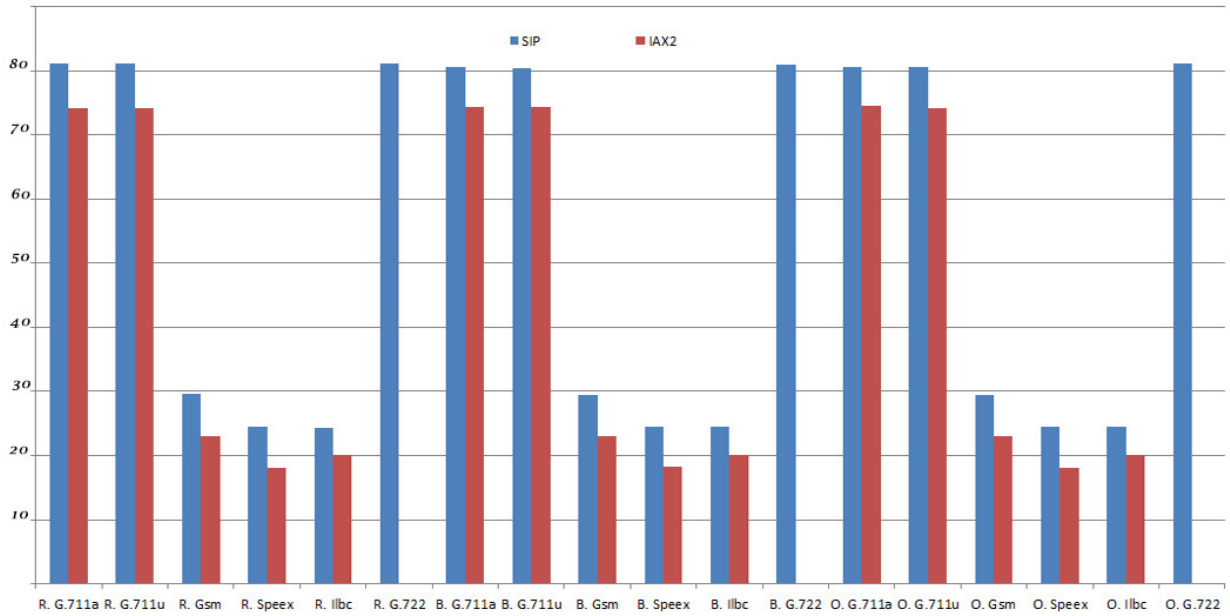
Todavia, a distinção é clara na abordagem dos protocolos SIP e IAX2, pois percebe-se que o protocolo IAX2 consome menor banda diante do protocolo SIP, conforme demonstra a figura 14. Para facilitar a identificação dos dispositivos na figura 14, foi inserido um código antecessor à nomenclatura dos CODEC'S, de modo que (R.) para o dispositivo Raspberry Pi 3, (B.) para o dispositivo Banana Pi M3 e (O.), respectivamente para o dispositivo Orange Pi Plus 2. Não foi possível realizar coleta do CODEC G.722 com o protocolo IAX2, por não encontrar *softphone* gratuito com suporte ao CODEC. Além disso, é possível afirmar que os desvios-padrão e o intervalo de confiança são mínimos, devido ao elevado número de amostras para realizar a abordagem.

Tabela 6 – Coleta Vazão IAX2. Unidade quilobits por segundo (Kbps).

Raspberry Pi 3 - Protocol IAX				
CODEC's	**Média**	**Desvio Padrão**	**Limite Inferior**	**Limite Superior**
G.711a	74,21	0,72	74,19	74,23
G.711u	74,20	0,72	74,18	74,22
Gsm	22,99	0,22	22,98	23,00
Speex	18,14	0,17	18,13	18,14
Ilbc	20,13	0,02	20,13	20,13
Banana Pi M3 - Protocol IAX				
G.711a	74,37	0,78	74,35	74,4
G.711u	74,33	0,78	74,31	74,36
Gsm	23,03	0,23	23,03	23,04
Speex	18,18	0,18	18,17	18,18
Ilbc	20,03	0,32	20,02	20,04
Orange Pi Plus 2 - Protocol IAX				
G.711a	74,57	0,70	74,55	74,60
G.711u	74,25	0,73	74,23	74,28
Gsm	23,01	0,23	23,00	23,02
Speex	18,16	0,18	18,15	18,16
Ilbc	20,13	0,19	20,12	20,13

Fonte: Autoria própria.

Figura 14 – Vazão SIP e IAX2. Unidade quilobits por segundo (Kbps).



Fonte: Autoria própria.

5.2.2 *Delay*

As tabelas 7 e 8 demonstram o *delay* dos pacotes coletados. É possível observar que o *delay* com o protocolo SIP encontra-se de acordo com a Rfc1890 (2017), a qual estabelece que o atraso padrão do pacote RTP deva ser 20ms. No entanto, há uma exceção do CODEC Ilbc, pois o mesmo destaca-se com um valor 50% superior aos demais, conforme se ilustra na figura 15. Isso ocorre devido ao seu algoritmo de codificação, que realiza elevada compressão do áudio e consequentemente eleva o atraso na transmissão.

Tabela 7 – Coleta *Delay* SIP. Unidade Milissegundos (ms).

Raspberry Pi 3 - Protocol SIP				
CODEC's	Média	Desvio Padrão	Limite Inferior	Limite Superior
G.711a	20,02	0,30	20,00	20,04
G.711u	20,02	0,30	20,00	20,04
Gsm	20,01	0,12	20,01	20,02
Speex	20,02	0,12	20,01	20,03
Ilbc	29,92	0,25	29,91	29,93
G.722	20,00	0,30	19,99	20,01
Banana Pi M3 - Protocol SIP				
G.711a	20,00	0,06	20,00	20,01
G.711u	20,00	0,05	20,00	20,01
Gsm	20,00	0,06	20,00	20,01
Speex	20,00	0,12	20	20,01
Ilbc	30,01	0,41	29,99	30,02
G.722	20,00	0,27	20,00	20,01
Orange Pi Plus 2 - Protocol SIP				
G.711a	20,00	0,06	20,00	20,01
G.711u	20,00	0,07	20,00	20,01
Gsm	20,00	0,11	20,00	20,01
Speex	20,00	0,12	20,00	20,01
Ilbc	30,01	0,17	30,00	30,01
G.722	20,00	0,10	20,00	20,01

Fonte: Autoria Própria

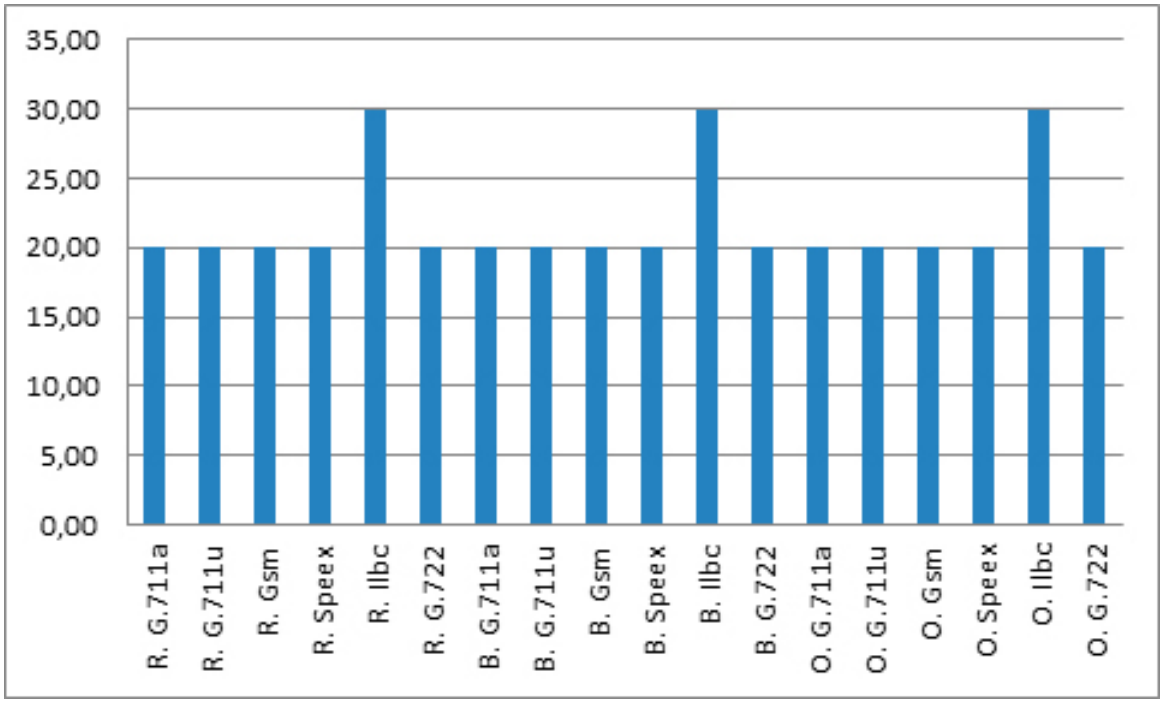
Já o *delay* com o protocolo IAX2 tende a zero com todos os CODEC'S abordados nesta pesquisa, o que valida a sua proposta como um novo padrão de protocolo. De acordo com Spencer, Guy e Miller (2010), trata-se de um protocolo aberto que realiza o transporte de sinalização e da mídia. Além disso, o protocolo IAX2 também possui como proposta eliminar quaisquer atrasos de transmissão.

Tabela 8 – Coleta *Delay* IAX2. Unidade Milissegundos (ms).

Raspberry Pi 3 - Protocol IAX				
CODEC's	Média	Desvio Padrão	Limite Inferior	Limite Superior
G.711a	0,02	0,00	0,02	0,02
G.711u	0,02	0,00	0,02	0,02
Gsm	0,01	0,02	0,02	0,02
Speex	0,02	0,00	0,02	0,02
Ilbc	0,03	0,00	0,03	0,03
Banana Pi M3 - Protocol IAX				
G.711a	0,02	0,01	0,02	0,02
G.711u	0,02	0,01	0,02	0,02
Gsm	0,02	0,01	0,02	0,02
Speex	0,02	0,01	0,02	0,02
Ilbc	0,03	0,01	0,03	0,03
Orange Pi Plus 2 - Protocol IAX				
G.711a	0,02	0,00	0,02	0,02
G.711u	0,02	0,00	0,02	0,02
Gsm	0,02	0,00	0,02	0,02
Speex	0,02	0,00	0,02	0,02
Ilbc	0,03	0,00	0,03	0,03

Fonte: Autoria Própria

Figura 15 – *Delay* SIP. Unidade Milissegundos (ms).



Fonte: Autoria própria.

5.2.3 *Jitter*

As tabelas 9 e 10 ilustram o *jitter* dos pacotes coletados. Diante dos dados apresentados, constata-se que tanto com o protocolo SIP quanto com o protocolo IAX2, o *jitter* tende a zero milissegundo. Isso se dá porque ambos os protocolos possuem como estratégia manter os quadros em um *buffer*, de modo a permitir que os quadros mais lentos cheguem a tempo para serem reproduzidos na sequência correta. Quanto maior a quantidade de *jitter*, maior será o número de quadros no *buffer*, a fim de minimizar ao máximo o *jitter* em chamadas VoIP.

Tabela 9 – Coleta *Jitter* Protocolo SIP. Unidade Milissegundos (ms).

Raspberry Pi 3				
CODEC's	**Média**	**Desvio Padrão**	**Limite Inferior**	**Limite Superior**
G.711a	0,15	0,09	0,15	0,15
G.711u	0,15	0,09	0,15	0,15
Gsm	0,10	0,23	0,09	0,10
Speex	0,21	0,15	0,21	0,21
Ilbc	0,24	0,15	0,23	0,25
G.722	0,15	0,10	0,14	0,15
Banana Pi M3				
G.711a	0,03	0,01	0,03	0,03
G.711u	0,03	0,01	0,03	0,03
Gsm	0,03	0,01	0,03	0,03
Speex	0,06	0,02	0,06	0,06
Ilbc	0,24	0,08	0,24	0,25
G.722	0,15	0,09	0,14	0,15
Orange Pi Plus 2				
G.711a	0,04	0,01	0,04	0,04
G.711u	0,04	0,01	0,04	0,04
Gsm	0,06	0,03	0,06	0,06
Speex	0,06	0,02	0,06	0,06
Ilbc	0,09	0,04	0,10	0,10
G.722	0,06	0,03	0,06	0,06

Fonte: Autoria Própria

Tabela 10 – Coleta *Jitter* Protocolo IAX2. Unidade Milissegundos (ms).

Raspberry Pi 3				
CODEC's	**Média**	**Desvio Padrão**	**Limite Inferior**	**Limite Superior**
G.711a	0,02	0,19	0,01	0,02
G.711u	0,02	0,19	0,01	0,02
Gsm	0,00	0,00	0,00	0,00
Speex	0,02	0,19	0,01	0,02
Ilbc	0,02	0,19	0,01	0,03
Banana Pi M3				
G.711a	0,02	0,19	0,01	0,03
G.711u	0,02	0,19	0,01	0,03
Gsm	0,02	0,19	0,01	0,02
Speex	0,02	0,19	0,01	0,02
Ilbc	0,03	0,19	0,01	0,02
Orange Pi Plus 2				
G.711a	0,02	0,19	0,01	0,02
G.711u	0,02	0,19	0,01	0,02
Gsm	0,02	0,19	0,01	0,02
Speex	0,02	0,19	0,01	0,02
Ilbc	0,02	0,19	0,01	0,02

Fonte: Autoria Própria

5.3 Chamadas Simultâneas e Eficiência Energética

Esta subseção aborda o quantitativo de chamadas simultâneas suportadas em cada dispositivo embarcado utilizado nesta pesquisa. Simultaneamente com a realização dos testes de chamadas, realizou-se também o monitoramento de memória RAM, processamento e energia, a fim de verificar o comportamento dos dispositivos durante elevado consumo de carga com o sistema de comunicação de voz sobre IP Asterisk.

Os testes foram realizados com o objetivo de alcançar o número máximo de chamadas suportadas com nenhuma ocorrência de erros. Assim, foram realizadas diversas tentativas até encontrar o resultado esperado. Em seguida, realizaram-se 10 repetições a fim de validar a ocorrência de nenhum erro. Trata-se de um resultado estático, pois não há variação da média e assim pode-se dispensar maior número de repetições (JAIN, 1991). Conquanto, foram necessárias aproximadamente 1000 execuções para consolidação desses resultados.

Para realizar a geração de carga de chamadas simultâneas SIP, utilizou-se o Asterisk em uma máquina virtual instalada no *notebook* e implementou-se um discador em *Shell Script*. Para discagens com o protocolo IAX2, bastou realizar a troca do canal no discador, de SIP para IAX2. O código do discador é apresentado na figura 16.

Com isso, dois arquivos foram gerados para realizar elevados números chamadas simultâneas, denominados CallFilePlay.sh e CallFileCodec.sh. O primeiro para chamadas normais, já

Figura 16 – Discador em *Shell Script*.

```
#!/bin/bash
aux1=$1;
aux2=$2;
CONTADOR=0
CONTADOR2=0
echo "Enviando $1 chamadas $2 vezes"

while [ $CONTADOR2 -lt $aux2 ]; do
echo "$CONTADOR2 x"
while [ $CONTADOR -lt $aux1 ]; do
echo "Channel: SIP/embarcado/5000" >>
/etc/asterisk/callfiles/teste$CONTADOR_$CONTADOR2.call
echo "CallerID: embarcado" >>
 /etc/asterisk/callfiles/teste$CONTADOR_$CONTADOR2.call
echo "MaxRetries: 0" >>
/etc/asterisk/callfiles/teste$CONTADOR_$CONTADOR2.call
echo "RetryTime: 60" >>
 /etc/asterisk/callfiles/teste$CONTADOR_$CONTADOR2.call
echo "WaitTime: 30" >>
/etc/asterisk/callfiles/teste$CONTADOR_$CONTADOR2.call
echo "Context: default" >>
/etc/asterisk/callfiles/teste$CONTADOR_$CONTADOR2.call
echo "Extension: 3000" >>
/etc/asterisk/callfiles/teste$CONTADOR_$CONTADOR2.call

mv /etc/asterisk/callfiles/teste$CONTADOR_$CONTADOR2.call
/var/spool/asterisk/outgoing/
let CONTADOR=CONTADOR+1;
done
sleep 10s;
CONTADOR=0
let CONTADOR2=CONTADOR2+1;
done
```

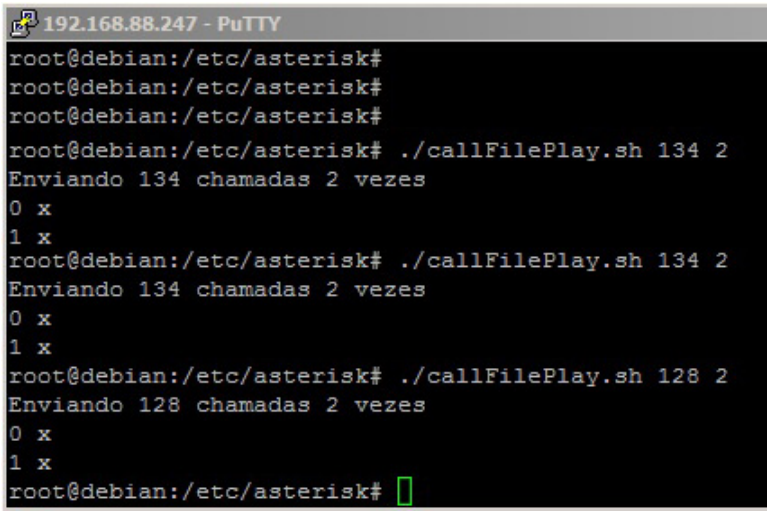
Fonte: Autoria própria.

o segundo para chamadas com transcodificação. A única diferença entre os dois arquivos é que no CallFileCodec.sh o número do ramal no discador é diferente. Desse modo, esses arquivos foram inseridos no diretório "/etc/Asterisk" do Asterisk, na máquina virtual discadora. E assim, foi possível iniciar os testes. A figura 17 mostra a aplicação do *script*.

Conforme ilustrado na figura 17, o comando ./callFilePlay.sh realiza a execução do *script*, já o número 128 significa a quantidade de chamadas a serem enviadas, e o 2 é o número de repetições a serem realizadas.

Também foi necessário configurar um arquivo de áudio denominado teste.gsm, com duração de 3 minutos, e incluí-lo no diretório de *sounds* "/var/lib/asterisk/sounds" do Asterisk,

Figura 17 – Utilização do arquivo CallFilePlay.sh.

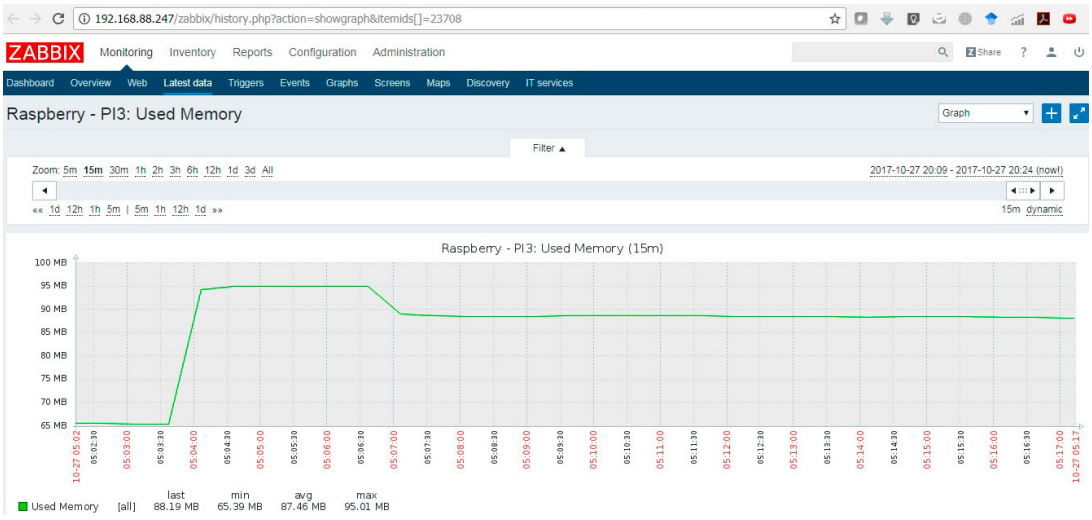


Fonte: Autoria própria.

cuja finalidade consiste em ao Asterisk receber as chamadas do gerador de carga, encaminhá-las para o autoatendimento e em seguida executar o áudio. Dessa forma, ocorre o fluxo de mídia por tempo limitado, o suficiente para a execução do teste de chamadas simultâneas.

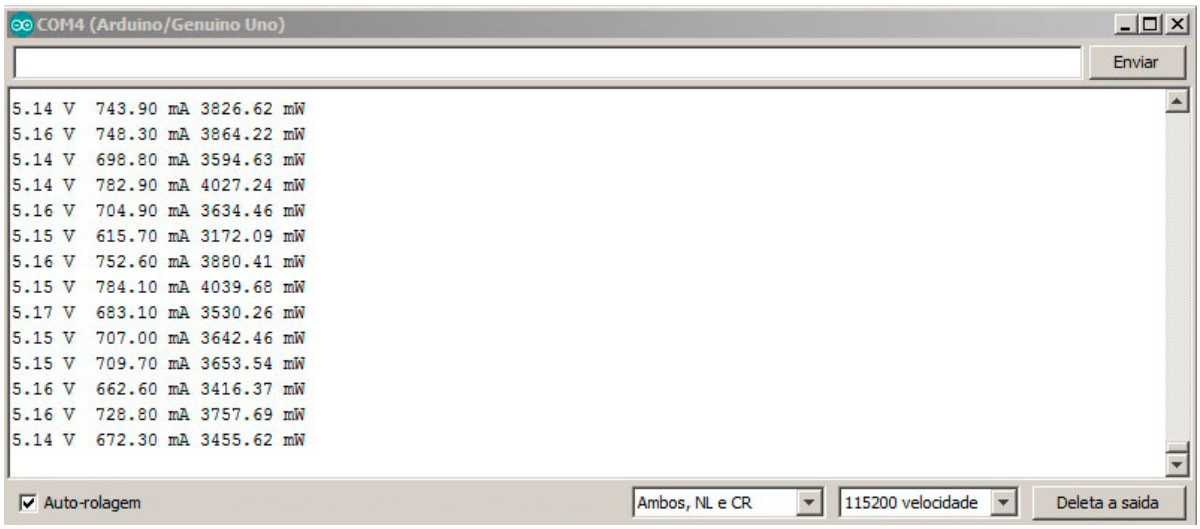
Para iniciar os testes, utilizou-se a máquina virtual no *notebook* para executar o discador gerador de chamadas, e em paralelo realizou-se o monitoramento do consumo de memória RAM e processamento com o *software* Zabbix; acompanhamento do consumo de energia com o circuito INA219, assim como verificação de ocorrência de erros no processamento das chamadas, através do *debug* no Asterisk. A figura 18 mostra o monitoramento da memória RAM no dispositivo Raspberry Pi 3, e a figura 19 mostra o monitoramento de energia realizado com o *software* do Arduino.

Figura 18 – Monitoramento de memória RAM.



Fonte: Autoria própria.

Figura 19 – Monitoramento de energia com INA219.

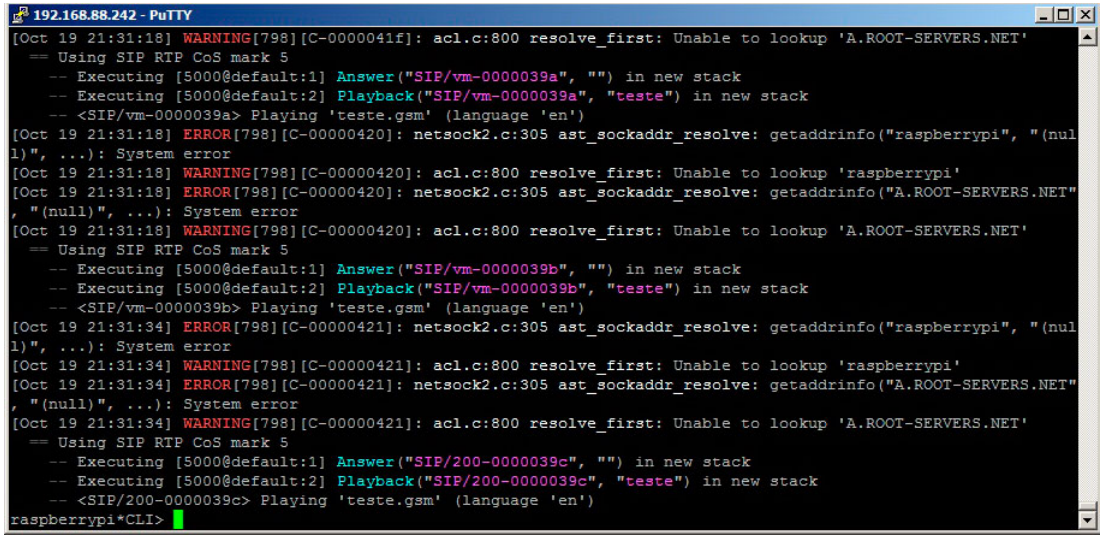


Fonte: Autoria própria.

Utilizou-se também um telefone IP Yalink T19p E2, a fim de validar a qualidade da chamada em momento de elevado consumo do sistema. Todavia, não foi possível validar a qualidade com o protocolo IAX2 e nem com os CODEC's Speex e Gsm com o protocolo SIP, devido as limitações de licença do aparelho telefônico.

Elevados fluxos de chamadas foram realizados entre o discador na máquina virtual e o Asterisk no dispositivo embarcado. De imediato, observou-se que o Asterisk do dispositivo embarcado só processava até 160 chamadas e o processamento permanecia baixo. Assim, iniciou-se uma investigação, a qual constatou a ocorrência de uma sequência de erros, conforme mostra a figura 20.

Figura 20 – Erro acima de 150 chamadas simultâneas.



Fonte: Autoria própria.

De acordo com a IBM (2017), esse problema está relacionado às limitações de usuário do Linux, e para corrigir utilizou-se o comando ulimit. No entanto, o dispositivo embarcado passou a processar apenas 200 chamadas simultâneas. Contudo, sem ocorrência de erros. Assim, verificou-se a configuração de limite do próprio Asterisk, e ao ampliá-la foi possível obter a quantidade máxima de chamadas que o dispositivo embarcado suporta.

À medida que os testes foram realizados, modificavam-se os CODEC's, assim como o protocolo. Os experimentos realizados foram divididos em 4 etapas, as quais serão expostas nas próximas subseções, assim como seus resultados.

5.3.1 Chamadas SIP

A tabela 11 ilustra os dados coletados com o protocolo SIP. Diante desses dados é possível observar que o CODEC GSM suportou o maior número de chamadas nos dispositivos Raspberry Pi 3 e Banana Pi M3. No entanto, há um elevado consumo de memória RAM. Todavia, diante da capacidade de memória RAM do dispositivo Raspberry Pi 3 de 1GB RAM e Banana Pi M3 de 2GB, esse resultado torna-se insignificante. Não obstante, o CODEC GSM possui o menor processamento. Já o CODEC G.711a se destaca por ser o CODEC que consome menos energia e suporta um número de chamadas significativo por se tratar de um dispositivo embarcado.

Tabela 11 – Dados coletados com o protocolo SIP.

Raspberry Pi 3 SIP							
CODEC's	**Repouso**	**Alaw**	**Ulaw**	**Gsm**	**Ilbc**	**Speex**	**G.722**
Chamadas simultâneas	0	257	251	291	211	211	241
Memória (MB)	48,48	95,01	93,92	121,13	102	94,66	125,75
Processamento (%)	0,05	64,36	70,55	55,72	100	100	78,89
Corrente (energia)	390,76	486,91	691,31	644,37	838,33	848,5	726,43
Banana Pi M3 SIP							
Chamadas suportadas	0	261	251	305	229	221	249
Memória (MB)	89,04	149,62	139,13	180,09	142,13	142,68	145,57
Processamento (%)	0,51	69,05	72,43	63,89	100	100	73,16
Corrente (energia)	686,11	890,39	950,45	934,21	1038,25	988,64	905
Orange Pi Plus 2 SIP							
Chamadas suportadas	0	247	261	251	241	241	251
Memória (MB)	81,97	113,77	125,5	113,51	122,32	124,46	117,83
Processamento (%)	0,56	98,97	99,82	99,48	98,67	98,53	99,34
Corrente (energia)	805,33	1133,69	1138,51	1119,26	1130,21	1135,64	1137,43

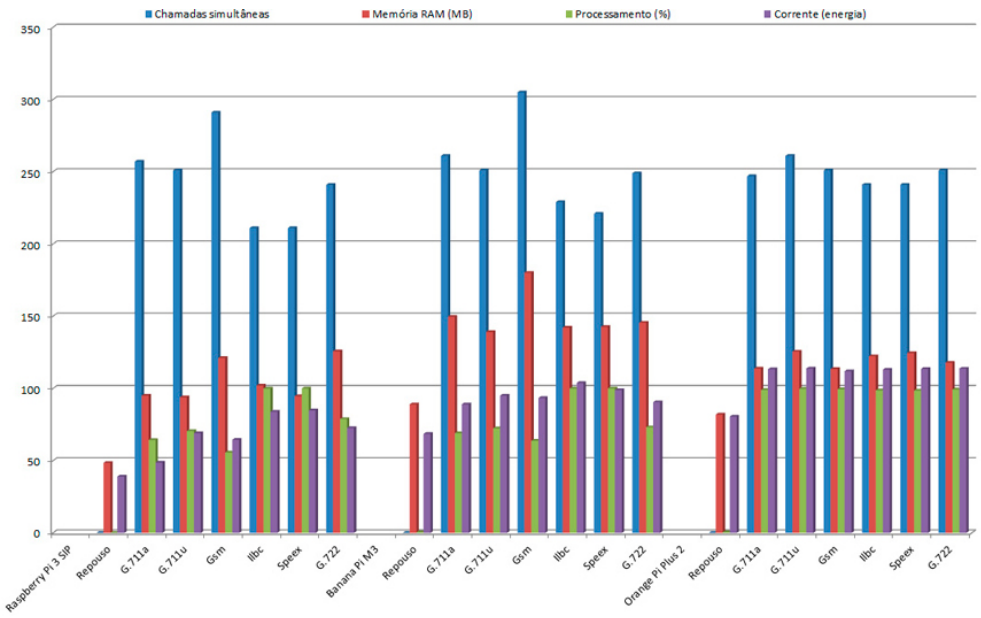
Fonte: Autoria Própria

Já o dispositivo Orange Pi Plus 2, obteve elevado processamento em todos os testes. Diante dessa informação, é possível afirmar que esse dispositivo não é ideal para uso com o Asterisk.

A figura 21 ilustra o comportamento das chamadas realizadas nos 3 dispositivos embarcados, a qual permite realizar melhor avaliação e comparação dos dados. Todavia, nota-se que o dispositivo Orange Pi Plus 2, possui resultados idênticos com todos os CODEC's abordados

nesta pesquisa, o que reforça a tese de que este dispositivo não serve para utilizar com o *software* Asterisk.

Figura 21 – Análise comparativa SIP.



Fonte: Autoria própria.

5.3.2 Chamadas SIP com Transcodificação

A tabela 12 demonstra os resultados coletados em chamadas realizadas com o protocolo SIP e com transcodificação. Nota-se que, com a transcodificação, o rendimento dos dispositivos diminui bastante. Mesmo assim o processamento do dispositivo Orange Pi Plus 2 permanece elevado.

Desse modo, mais uma vez reforça a afirmação em que esse dispositivo não é adequado para utilizar o Asterisk. Já nos demais dispositivos observa-se que os CODEC's G.711a e G.711u possuem maior número de chamadas simultâneas suportadas e menor consumo de processamento, memória e energia.

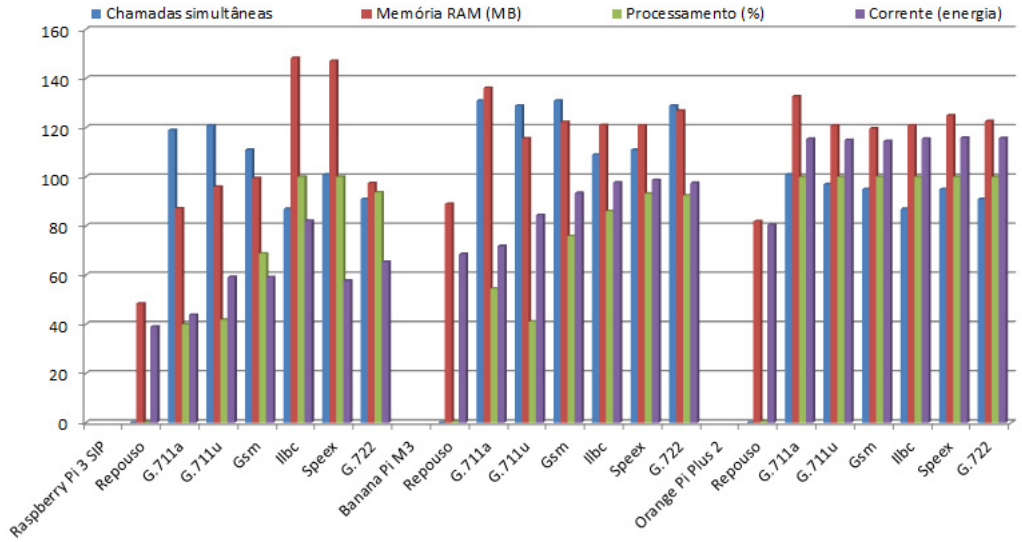
Além disso, o dispositivo Raspberry Pi 3 destaca-se com um considerável consumo de processamento com os CODEC's G.711a e G.711u, conforme ilustra a figura 22.

Tabela 12 – SIP com Transcodificação.

Raspberry Pi 3							
CODEC's	**Repouso**	**Alaw**	**Ulaw**	**Gsm**	**Ilbc**	**Speex**	**G.722**
Chamadas simultâneas	0	119	121	111	87	101	91
Memória (MB)	48,48	87,17	95,98	99,47	148,3	247,16	97,49
Processamento (%)	0,05	3,97	41,87	68,83	100	100	93,7
Energia (corrente)	390,76	438,85	592,64	592,04	821,49	577,56	654,27
Banana Pi M3							
Chamadas suportadas	0	131	129	131	109	111	129
Memória (MB)	89,04	136,14	115,63	122,3	121,01	120,95	126,86
Processamento (%)	0,51	54,57	41,13	75,87	86,11	93,23	92,48
Energia (corrente)	686,11	718,81	844,7	934,56	977,82	987,33	975,25
Orange Pi Plus 2							
Chamadas suportadas	0	101	97	95	87	95	91
Memória (MB)	81,97	132,74	120,92	119,7	120,95	124,95	122,67
Processamento (%)	0,56	100	99,99	99,99	99,98	99,99	99,99
Energia (corrente)	805,33	1154,52	1150,26	1145,42	1155,57	1159	1157,58

Fonte: Autoria Própria

Figura 22 – Análise comparativa SIP com Transcondificação.



Fonte: Autoria própria.

5.3.3 Chamadas IAX2

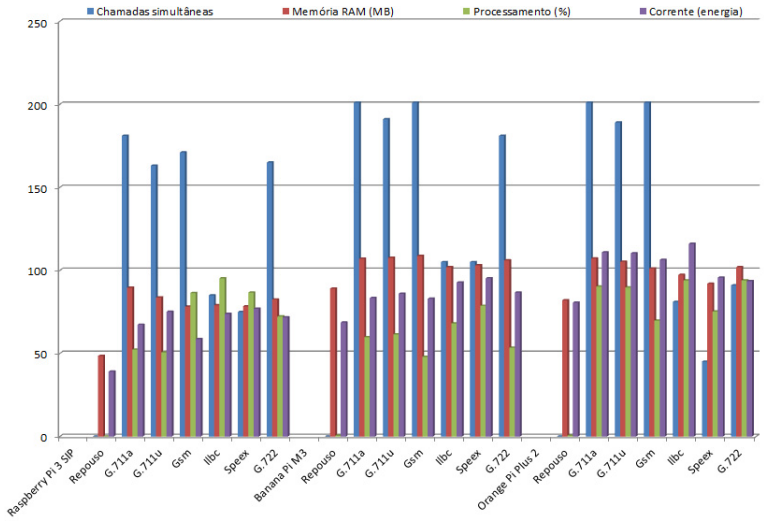
Esta subseção aborda os dados coletados com o uso do protocolo IAX2. Os mesmos são ilustrados na tabela 13. Observa-se que o CODEC G711.a se destaca com baixo consumo de memória, processamento, energia e elevado número de chamadas simultâneas, nos dispositivos Raspberry Pi 3 e Banana Pi M3. Já o Orange Pi Plus 2 obteve como destaque o CODEC GSM devido ao elevado número de chamadas simultâneas, baixo processamento, consumo de energia e moderado consumo de memória. Mas, se comparado aos demais dispositivos abordados nesta pesquisa, esse é o que possui desempenho inferior conforme mostra a figura 23.

Tabela 13 – Dados coletados com o protocolo IAX2.

Raspberry Pi 3 IAX2							
Codec's	Repouso	Alaw	Ulaw	Gsm	Ilbc	Speex	G.722
Chamadas simultâneas	0	181	163	171	85	75	165
Memória (MB)	48,48	89,54	83,66	78,16	78,99	78,38	82,49
Processamento (%)	0,05	52,28	50,67	86,41	95,21	86,64	72,23
Energia (corrente)	39,076	67,284	75,122	58,587	73,816	76,871	71,732
Banana Pi M3 IAX2							
Chamadas suportadas	0	201	191	201	105	105	181
Memória (MB)	89,04	107,09	107,62	108,77	101,94	103,04	106,08
Processamento (%)	0,51	59,82	61,47	47,97	68,06	78,69	53,46
Energia (corrente)	68,611	83,392	85,964	82,92	92,7	95,299	86,695
Orange Pi Plus 2 IAX2							
Chamadas suportadas	0	201	189	201	81	45	91
Memória (MB)	81,97	107,23	105,25	101,09	97,29	91,97	102
Processamento (%)	0,56	90,3	89,77	69,78	94	75,27	94
Energia (corrente)	80,533	110,878	110,31	106,29	116,047	95,658	93,499

Fonte: Autoria própria

Figura 23 – Análise comparativa IAX2.



Fonte: Autoria própria.

5.3.4 Chamadas IAX2 com Transcodificação

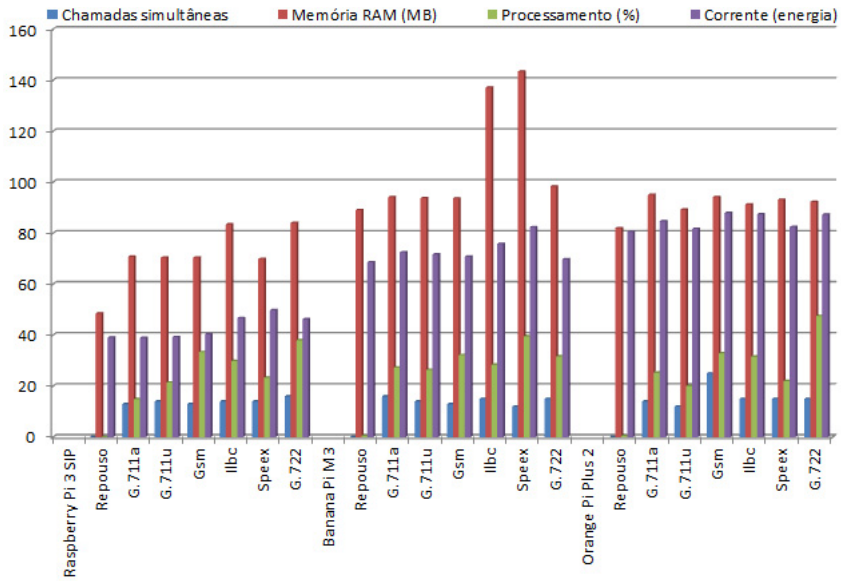
A tabela 14 apresenta dados coletados com o uso do protocolo IAX2 com transcodificação. De imediato, verifica-se um baixo número de chamadas simultâneas suportadas. Diante desse contexto, é possível afirmar que os dispositivos embarcados abordados nesta pesquisa não dão suporte ao processo de transcodificação de CODEC's com o protocolo IAX2. Apenas nesse teste o dispositivo Orange Pi Plus 2 não reportou elevado consumo de processamento, conforme ilustra a figura 24.

Tabela 14 – Dados coletados com o protocolo IAX2 com Transcodificação.

Raspberry Pi 3							
Codec's	**Repouso**	**Alaw**	**Ulaw**	**Gsm**	**Ilbc**	**Speex**	**G.722**
Chamadas simultâneas	0	13	14	13	14	14	16
Memória (MB)	48,48	70,81	70,43	70,43	83,48	69,95	84,02
Processamento (%)	0,05	14,98	21,46	33,35	29,92	23,3	37,99
Energia (corrente)	390,76	389,51	392,62	403,92	467,55	498,12	463
Banana Pi M3							
Chamadas suportadas	0	16	14	13	15	12	15
Memória (MB)	89,04	94,18	93,79	93,66	137,07	143,37	98,33
Processamento (%)	0,51	27,34	26,44	32,18	28,42	39,57	31,74
Energia (corrente)	686,11	725,73	716,77	708,26	758,22	823,97	697,62
Orange Pi Plus 2							
Chamadas suportadas	0	14	12	25	15	15	15
Memória (MB)	81,97	95,12	89,29	94,15	91,29	93,15	92,26
Processamento (%)	0,56	25,32	20,06	32,92	31,63	22,05	47,52
Energia (corrente)	805,33	846,61	817,15	879,85	874,27	825,3	873,12

Fonte: Autoria própria

Figura 24 – Análise comparativa IAX2 com transcodificação.



Fonte: Autoria própria.

6
Conclusão

Neste trabalho, foi realizado uma abordagem para análise de desempenho e eficiência energética em três dispositivos embarcados de última geração com uso do sistema de comunicação por voz sobre IP Asterisk, o qual aferiu o *jitter*, *delay* e *vazão* com os protocolos SIP e IAX2 com os CODEC's G.711a (Alaw), G.711u (Ulaw), G.722, Ilbc, Speex e Gsm.

As medições foram realizadas com a finalidade de comparar os três dispositivos embarcados com uso do Asterisk. No entanto, os resultados demonstraram grande similaridade nos dados, tanto com o protocolo SIP, quanto com o protocolo IAX2, isso no requisito *network*.

Verificou-se também o quantitativo de chamadas simultâneas suportadas em cada dispositivo com os protocolos SIP e IAX2, tanto em chamadas normais quanto em chamadas transcodificadas e em paralelo, realizou-se a análise do consumo de memória RAM, processamento e energia.

A prototipação foi realizada para comparar os três dispositivos embarcados com uso do Asterisk. Os resultados foram significativos. Os dispositivos Raspberry Pi 3 e Banana Pi M3 suportam de forma satisfatória um elevado número de chamadas simultâneas com consumo de memória, processamento e energia moderado através do CODEC G.711a e G.711u. No entanto, o dispositivo Orange Pi Plus 2 demostrou alto consumo de processamento. Assim, é possível afirmar que esse dispositivo não é adequado para uso do Asterisk.

Todos os 3 dispositivos apresentaram estabilidade durante toda a pesquisa. Não ocorrendo nenhuma reinicialização involuntária dos equipamentos, mesmo durante o fluxo de elevadas cargas.

6.1 Trabalhos Futuros

O desempenho dos 3 dispositivos embarcados abordados neste trabalho foram avaliados com a finalidade de encontrar o melhor dispositivo para suportar o sistema de comunicação por voz sobre IP Asterisk. Não obstante, novos dispositivos embarcados, assim como novas tecnologias, surgirão. Dessa forma, amplia-se a possibilidade de estender este trabalho.

Como trabalhos futuros, propõe-se a realização desses experimentos com o CODEC Opus, uma vez que não foi possível realizar a compilação nos dispositivos embarcados devido à incompatibilidade. Pode-se também realizar a execução dessa mesma abordagem com protocolo IAX2, com uso de criptografia. É possível ainda propor uma comparação do comportamento do protocolo IAX2 com transcodificação em plataformas computacionais com arquitetura x86. Bem como realizar os mesmos testes de vazão, *jitter* e *delay* com outras ferramentas disponíveis no mercado, específicas para análise de pacotes VoIP, como NetQuality Voip e SIP Tester.

Além das técnicas de medições apresentadas é possível acoplar novas técnicas e assim substituí-las ou até mesmo aperfeiçoá-las. Realizar testes com técnicas diferentes é importante para definir a técnica que possui melhor resultado.

Referências

ABID, F. et al. Embedded implementation of an IP-PBX /VoIP gateway. *Proceedings of the International Conference on Microelectronics, ICM*, n. Icm, p. 5–8, 2012. ISSN 2159-1660. Citado 5 vezes nas páginas 13, 14, 26, 27 e 30.

ANDROULIDAKIS, I. I. *VoIP and PBX Security and Forensics*. [s.n.], 2016. ISSN 2191-8112. ISBN 978-3-319-29720-0. Disponível em: <http://link.springer.com/10.1007/978-3-319-29721-7>. Citado na página 12.

Banana Pi. *Banana Pi M3*. 2017. Disponível em: <http://www.banana-pi.org/m3.html>. Acesso em: 01 nov. 2017. Citado na página 24.

BERNAL, P. S. M. *Voz sobre protocolo IP: A nova realidade da telefonia*. São Paulo: [s.n.], 2007. 271–350 p. ISBN 978-85-365-0174-1. Citado na página 12.

BERTOLOTTI, I. C.; HU, T. *Embedded Software Development: The Open-Source Approach*. San Francisco: [s.n.], 2015. ISBN 9781466593930. Citado na página 21.

BRYANT, R.; MADSEN, L.; MEGGELEN, J. V. *Asterisk: The Definitive Guide*. 4. ed. Sebastopol: [s.n.], 2013. ISBN 9781449332426. Citado 2 vezes nas páginas 12 e 19.

COLCHER, S. et al. *VoIP: voz sobre IP*. Rio de Janeiro: [s.n.], 2005. Citado na página 12.

Dalle Vacche, A.; Kewan Lee, S. *Zabbix Network Monitoring Essentials*. Birmingham: [s.n.], 2015. 178 p. ISBN 978-1784399764. Citado na página 39.

DEON, S. *VoIP et ToIP Asterisk: La téléphonie IP d'entreprise*. 2. ed. SAINT HERBLAIN: [s.n.], 2010. v. 2. 421 p. ISBN 2746058065. Citado na página 17.

DEY, N.; MUKHERJEE, A. *Embedded Systems and Robotics with Open Source Tools*. Boca Raton: [s.n.], 2016. 121 p. ISBN 978-1-4987-3440-0. Citado na página 21.

DIGIUM. *Dimensioning*. 2017. Disponível em: <http://www.digium.com/blog/2012/09/25/asterisk-dimensioning-what-server-do-i-need/>. Acesso em: 10 nov. 2017. Citado na página 22.

EDAN, N. M. et al. Performance evaluation of QoS using SIP & IAX2 VVoIP protocols with CODECS. *Proceedings of 2016 SAI Computing Conference, SAI 2016*, p. 631–636, 2016. Citado 6 vezes nas páginas 13, 14, 28, 30, 34 e 35.

FLANAGAN, W. A. *VoIP and Unified Communications*. Hoboken: [s.n.], 2012. ISBN 9781118166048. Disponível em: <http://doi.wiley.com/10.1002/9781118166048>. Citado na página 19.

GONÇALVES, F. E. *Asterisk Configuration Guide*. Florianópolis: [s.n.], 2007. ISBN 9788590690429. Citado 3 vezes nas páginas 13, 19 e 21.

HARTPENCE, B. *Packet Guide to Voice over IP*. Sebastopol: [s.n.], 2013. 242 p. ISBN 9781449339678. Citado na página 17.

IBM. *Diretrizes para configurar as configurações de ulimit do Linux*. 2017. Disponível em: <https://www.ibm.com/support/knowledgecenter/pt-br/SSCRJU{_}4.0.0/com.ibm.streams.install.doc/doc/ibminfospherestreams-install-operating-system-settings.h>. Acesso em: 20 out. 2017. Citado na página 53.

JAIN, R. *The Art of Computer Systems Performance Analysis: Techniques for Experimental Design, Measurement, Simulation, and Modeling*. New York: [s.n.], 1991. ISBN 978-0471503361. Citado 2 vezes nas páginas 32 e 49.

JOHNSTON, A. B. *SIP : Understanding the Session Iniation Protocol*. 3. ed. Boston: [s.n.], 2009. 427 p. ISBN 9781607839958. Citado na página 20.

KELLER, A. *Asterisk na prática*. São Paulo: [s.n.], 2011. ISBN 978-85-7522-286-7. Citado 5 vezes nas páginas 16, 17, 18, 19 e 20.

KELLY, T. *VoIP for Dummies*. Indianapolis: [s.n.], 2005. 313 p. ISBN 978-0-7645-8843-3. Citado na página 16.

LOVERPI. *Comparison*. 2017. Disponível em: <https://www.loverpi.com/blogs/news/94801153-raspberry-pi-3-banana-pi-m3-orange-pi-plus-2-odroid-c2-spec-comparison>. Acesso em: 10 nov. 2017. Citado na página 22.

MAHLER, P. *VoIP Telephony with Asterisk*. Greenville: [s.n.], 2005. 247 p. ISBN 0975999206. Citado 2 vezes nas páginas 18 e 19.

MAIA, W. P. *PROJETO, IMPLEMENTAÇÃO E DESEMPENHO DOS ALGORITMOS CRIPTOGRÁFICOS AES, PRESENT E CLEFIA EM FPGA*. Aracaju: Universidade Federal de Sergipe, 2017. Disponível em: <https://ri.ufs.br/bitstream/riufs/5029/1/WILLIAM{_}PEDROSA{_}MA>. Citado 2 vezes nas páginas 35 e 36.

Mohamed Boucadair. *Inter-Asterisk Exchange (IAX): Deployment Scenarios in SIP-Enabled Networks*. West Sussex: [s.n.], 2009. ISBN 9780470770726. Citado 2 vezes nas páginas 20 e 21.

Orange Pi Plus 2. *Orange pi*. 2017. Disponível em: <http://www.orangepi.org/orangepiplus2/>. Acesso em: 01 nov. 2017. Citado na página 24.

OUAKIL, L.; PUJOLLE, G. *Téléphonie sur IP*. 2. ed. Versalhes: [s.n.], 2008. 484 p. ISBN 9782551238224. Citado 2 vezes nas páginas 17 e 19.

RFC1890. *RTP Profile for Audio and Video Conferences with Minimal Control*. 2017. Disponível em: <https://tools.ietf.org/html/rfc1890>. Acesso em: 15 nov. 2017. Citado na página 46.

RFC5456. *Rfc5456*. 2016. Disponível em: <https://tools.ietf.org/html/rfc5456>. Acesso em: 20 n20.nov16. Citado 2 vezes nas páginas 20 e 21.

SHIMONSKI, R. *Wireshark - Guia Prático - Análise e Resolução de Problemas de Tráfego de Rede*. São Paulo: [s.n.], 2014. ISBN 9788575223888. Citado na página 37.

SIEWERT, S.; PRATT, J. *REAL-TIME EMBEDDED COMPONENTS AND SYSTEMS with LINUX and RTOS*. Boston: [s.n.], 2016. v. 1. ISBN 9781942270041. Citado na página 21.

SINNREICH, H.; JOHNSTON, A. B. *Internet communications using SIP: Delivering VoIP and multimedia services with Session Initiation Protocol*. 2. ed. Indianapolis: Wiley Publishing, Inc., 2006. v. 27. 408 p. ISBN 0470067861. Citado na página 20.

SPENCER, M.; GUY, E.; MILLER, F. RFC5456. p. 1–101, 2010. Citado na página 46.

SULKIN, A. *PBS Systems For Ip Telephony: Migrating Enterprise Communications*. Sykesville: [s.n.], 2002. Citado na página 12.

SULTAN, P. *Asterisk: La Telephone d'enterprise libre*. Versalhes: [s.n.], 2009. ISBN 9782212124347. Citado na página 17.

TESFAMICAEL, A. D. et al. Implementation and Evaluation of Open Source Unified Communications for SMBs. *2014 International Conference on Computational Intelligence and Communication Networks*, p. 1243–1248, 2014. Disponível em: <http://ieeexplore.ieee.org/document/7065677/>. Citado 2 vezes nas páginas 29 e 30.

VILLACÍS, D.; ACOSTA, F. R.; Lara Cueva, R. A. Performance analysis of VoIP services over WiFi-based systems. *2013 IEEE Colombian Conference on Communications and Computing, COLCOM 2013 - Conference Proceedings*, p. 1–6, 2013. Citado 3 vezes nas páginas 27, 30 e 34.

VOIP-INFO. *Asterisk*. 2017. Disponível em: <http://www.voip-info.org/wiki/view/Asterisk>. Acesso em: 16 jan. 2017. Citado 2 vezes nas páginas 13 e 42.

WHITE, E. *Making Embedded Systems*. Sebastopol: O'Reilly Media, Inc., 2011. 330 p. ISBN 9781449302146. Citado na página 21.

YEUNG, J. *VOIP - A practical guide for the non-telephone engineer*. 2015. Citado na página 17.

ZAPALS. *Raspberry Pi 3 Model B Motherboard*. 2017. Disponível em: <https://www.zapals.com/raspberry-pi-3-model-b-motherboard-on-board-wi-fi-bluetooth-development-board-rs-original-uk-version.html>. Acesso em: 01 nov. 2017. Citado na página 23.

Apêndices

APÊNDICE A – WEBIST 2017 - B3

A Security Approach using SIP Protocol in Imbedded Systems

Toniclay Andrade Nogueira, Adauto Cavalcante Menezes,
Admilson De Ribamar Lima Ribeiro and Edward David Moreno Ordonez
Computing Department, Segipe Federal University, Aracaju, Brazil

Keywords: VoIP, Asterisk, Embedded Systems, SIP, Safety.

Abstract: Voice over IP communication will dominate the world. However, given the growing demand for voice and data communication to make any and all communication reliable and secure, several attacks occur frequently in communication networks, so this work is based on verifying security, analyzing risks, vulnerabilities, such as verifying the attacks and proposing a security measure for voice over IP communication on embedded devices.

1 INTRODUCTION

The Voice Over Internet Protocol (VoIP) technology consists in the integration of the services in the telecommunication areas and the network services provided by computers. In this way, it enables the digitization and encoding of the voice signal and transforms it into data packages for communication in a network using UDP protocols.

In this context, the VoIP concept allows cost reduction in installations, maintenance and management of parallel networks. With this, a new concept of telephony is created (Sitolino 1999). However, it will be necessary equipment, techniques and specific human resources (Silva, 2016).

Stapko (2007) understands as information security the protection of personal or confidential information, as well as the computational resources of individuals or organizations. Without information security, malicious individuals can destroy or use such information for malicious purposes. The state-of-the-art security in VoIP telephony involves audio encryption between the two distinct points as well as interoperability between communication server manufacturers through indecipherable encryption and centralized management (Stallings, 2008).

According to Barr (1999), Car and Wagner (2003) and Marwedel (2011), embedded systems must be reliable, since failures can compromise their functionality and make system recovery unfeasible. Embedded systems use hardware platforms, which are driven by softwares. Several implementations of processors can be used, which implies a great reduction of costs.

Some reliability issues are found on embedded devices, as they cannot be safely shut down for repairs, the system must run continuously.

As its operating mode has reduced performance, the environment tends to fail if it is turned off (Akyildiz, 2002). Security in embedded systems was not always considered, since most of these systems were initially operated without Internet connectivity.

Information security and new embedded device paradigms are increasingly present in our lives. However, the communication between devices will have a great impact on global communication, which will increase the efficiency and security of VoIP communication.

This article is organized as it follows: section 2 is composed of theoretical grounding presentation, section 3 presents related works, section 4 has a description of the proposal, and section 5 show the expected contribution of the research.

2 THEORETICAL FOUNDATION

2.1 VoIP

According to Raake (2007) and Walker, (2004), VoIP is a technology that performs voice communication over an IP network.

The communication process consists of transforming the analog voice into digital, through the

Nogueira, T., Menezes, A., Ribeiro, A. and Ordonez, E.
A Security Approach using SIP Protocol in Imbedded Systems.
DOI: 10.5220/0006354403520355
In *Proceedings of the 13th International Conference on Web Information Systems and Technologies (WEBIST 2017)*, pages 352-355
ISBN: 978-989-758-246-2

fragmentation of the package and transport over the IP network. The process is becoming more modern, it is possible to mention some softwares that work with this technology, among them, Facebook, Messenger, Skype, Viber and WhatsApp.

In image 01 we can observe the operation of a VoIP application where the analog audio is converted into digital and grouped into packages that are transmitted to the IP network through the Real Time Protocol (RTP) protocol, after arriving at the receiver the packages are organized and then reproduced.

Figure 1: Scenario of the ideal operation of the VoIP application, Shigueoka, 2016.

2.2 Embedded Systems

Some data researched in high technology shows that more than 90% of microcomputers manufactured in the world are intended for machines that are not called computers, such as cell phones, automobiles, DVD players, among others.

According to Reis (2004), what comes to differentiate the set of devices from a computer is the project based on a dedicated set and specialist, consisting of Hardware, Software and Peripherals, i.e., embedded system.

For Ball (2005), the system is classified as embedded when it is dedicated to a single task and continuously interacts with the environment around it, by the use of actuators and sensors.

In their article, Siqueira, Menegotto, Weber, César Netto and Wagner (2006) comment on the use of embedded systems in critical applications, which comprises as applications in which the risks associated with the hazards involved are considered unacceptable and need to be handled.

The embedded system is commonly a solution formed from dedicated and specific microcontroller and software to performing the operational functions of equipment for which it was designed.

2.3 Session Initiation Protocol (SIP)

SIP has been developed to facilitate the implementation of the basic aspects of a session, which is a non-trivial process. Today it is used worldwide and it is also a strong “competitor” of H.323. Barbosa (2006) defines SIP as a protocol that signals client-server sessions, and that stands out for its simplicity and mobility; it has a primitives the initialization, modification and termination of sessions in a VoIP communication.

According to Defsip, together with Real-time Transport Protocol (RTP), Real Time Streaming Protocol (RTSP), Session Description Protocol (SDP), SIP establishes a complete multimedia architecture, providing complete services to the user.

SIP also provides participant management services in a multimedia session.

According to Cuervo (2000), due to the ability of working in conjunction with other protocols, it allows integration with public telephony, allowing not only the connection between IP extensions, but also for public network telephones.

2.4 Types of Attacks to SIP Protocol

2.4.1 Main-in-the-middle

For this attack, the attacker can use two techniques: ARP table poisoning, or DNS cloning. With either of these, permission is granted to be between the SIP server and the User Agent. With this type of attack, the intruder does not necessarily know valid usernames and passwords; they can simply route traffic between the server the and client and act intercepting the packages, preventing them from reaching their real destination, which is the SIP server (Nakamura, Emílio, Geus and Lício, 2007).

2.4.2 Subsection Titles

According to Thermos (2007), this attack aims to obtain credentials from valid users in a SIP telephony communication system using a brute-force attack, which is, sending multiple ID requests and passwords to from a dictionary.

2.4.3 Denial of Service

In attacks known as Denial of Service, it is possible to layers of infrastructure in VoIP environment. According to Thermos (2007) the DoS attacks have as main objective to cause the interruption of the target service. In this case, the attack is directed to both the operating system and also to the network services.

2.4.4 SIP Redirect

For Butcher, Li and Guo (2007), the attack employs a server that receives requests from a telephone or proxy and returns a redirect response, which indicates

where the request is to be repeated, thus enabling users to have a call redirected to another location rather than where they are located. However, the caller normally dials only the number to reach the user.

The attacker redirects the victim's calls to a specific number, so the attacker starts receiving the calls that were forwarded to the attacked user.

3 RELATED WORK

The work covered in this article contains a large amount of research. Thus, the IEEE Base has been used with works from the year 2012 to 2016.

3.1 The Communication System

In their article, Lomotey and Deters (2014) show that IP communication systems have been the target of attacks, such as call theft, attacks on servers, which allows access to users' data. Thus, the author proposed a solution to prevent the intrusion of attackers in the communication system built in VoIP Asterisk.

In his experiment, a complete platform for Asterisk was not used because he proposed a cloud-based middleware, which layer maintains the most sensitive part of the information call.

3.2 VoIP Security Analyses

Rehman and Abbasi (2014) analyzed security in the VoIP architecture with the Asterisk voice over IP communication system. It has been noted that most of the attacks are related to the fragility of the SIP protocol, espionage attacks, modification and involuntary interruption were detected.

The authors have proposed as solution of the presented problem, an efficient and secure mechanism of authentication for the protocol SIP, with this, it is possible to assure greater protection to the attacks.

It was suggested to assign a cryptographic token that would authenticate the users allowing their identification and providing greater security as well as there would be the need for the user to enter a password to use other available services.

3.3 VoIP Intrusion Detection with Snort

Číž, Lábaj, Podhradský and Londák (2012), have proposed in their experiment a model focused on DoS attack in order to cause a malfunction in Asterisk voice over IP communication software. The authors used the SIP tool, in order to verify the functionality of the detection system and cause anomalies in denial of service attacks, the Snort software tool was also used to detect open network attacks, capable of performing analysis of the Traffic and packet logging on IP networks.

4 DESCRIPTION OF THE PROPOSAL

The present proposal aims at creating a defense method for the IP asterisk over SIP protocol, in order to use embedded devices (Raspberry Pi3, Banana Pi M3 and Orange Pi Plus 2) as shown in image 2. The method will be based on the main attacks that occur in embedded systems, contemplated by authors in related works in diverse bases.

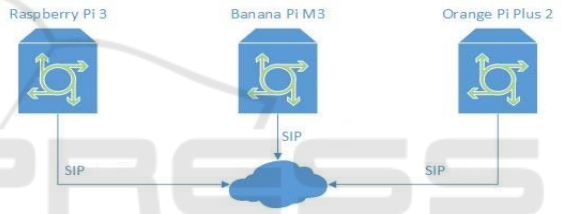


Figure 2: Architecture of the scenario.

With the result of the main attacks, simulations will be made in the device in order to propose security methods.

It will be necessary to study the SIP protocol to verify the vulnerabilities in order to apply the best configuration and defense methods to ensure the security of the device.

In order to achieve the objectives of this research, it will be necessary to elaborate a scenario that makes possible to carry out all the experiments as close as possible to a real production environment, so the scenario should include three low cost embedded devices already configured with the system, which must be directly connected to the Internet.

5 EXPECTED CONTRIBUTION

This proposal presents as main contribution the elaboration of a security method for a VoIP communication central in an embedded device using the Asterisk system.

With the development of this proposal we intend

to obtain the following contributions: to survey the main techniques used to attack the communication systems, to survey the tools and materials necessary to simulate the most significant attacks on embedded devices; to perform a literature review of the SIP protocol, to analyze the vulnerabilities of the SIP protocol, to propose a defense for these attacks, to write the dissertation and present the results of the security analysis on the devices shipped with Asterisks.

REFERENCES

Akyildiz, I. F., Su, W., Sankarasubramaniam, Y., and Cayirci, E. (2002). Wireless sensor networks: a survey. *Computer networks, 38(4):393–422.*

Ball, Stuart - ”Embedded Microproceessor Systems: Real Woard Desing”, *3º edition,* Editora:Mcpros, EUA, 2005.

Barbosa, Camila Soares. *Voz sobre IP em Redes Locais sem Fio*. CEFET – Centro Federal de Educação Tecnológica de Goiás. Goiânia, (2006).

Barr, M. (1999). *Programming embedded systems in C and C++*. O'Reilly.

Carro, L. and Wagner, F. R. (2003). Sistemas computacionais embarcados. *Jornadas de atualização em informática. Campinas:* UNICAMP.

Číž, P., Lábaj, O., Podhradský, P., & Londák, J. (2012). VoIP Intrusion Detection System with Snort. *In ELMAR, 2012 Proceedings (pp. 137-140).* IEEE.

Cuervo, F., Greene, N., Rayhan, C., Rosen, B., and Segers, J. (2000). *Megaco Protocol Version 1.0 RFC 3015 (Proposed Standrd)*. Obsoleted by RFC 3525.

D. Butcher, X. Li, and J. Guo. Security challenge and defense in VoIP infrastructures. *IEEE Transactions on Systems, Man, and Cybernetics, Part C:* Applications and Reviews, 37(6):1152–1162, November 2007.

DEFSIP. *Definindo o que é um protocolo de sinalização*. Disponível em http://www.gta.ufrj.br/grad/06_1/SIP/Definindooqueumprotocolodesinalizao.html, acessado em 27 nov 2016.

Lomotey, R. K. and DETERS, R. (2014). Intrusion Prevention in Asterisk-Based Telephony System. In 2014 *IEEE International Conference on Mobile Services, pages 116–123*.IEEE.

Marwedel, P. (2011). Embedded system design. Springer.

Nakamura, Emílio T.; *GEUS, Paulo Lício de. Segurança de rede em ambientes corporativos.* São Paulo: Novatec Editora, 2007.

Raake, *A. Speech quality of VoIP: assessment and prediction. [S.l.]:* John Wiley & Sons, 2007.

Rehman, U. U. and Abbasi, A. G. (2014). Security analysis of VoIP architecture for identifying SIP vulnerabilities. *In 2014 International Conference on Emerging Technologies (ICET), number i, pages 87–93*. IEEE.

Reis, Claiton – *"Sistemas Operacionais para Sistemas Embarcados",* Tutorial, Editora: EDUFBA, BRASIL, 2004.

Silva, Adailton. Qualidade de Serviço em VoIP – *Rede Nacional de Ensino e Pesquisa.* Maio/2000 – Disponível em: http://www.rnp.br/newsgen - Acessado em 18/11/2016.

Siqueira, Tórgan Flores de ; Menegotto, C. C. ; Weber, T. S. ; César Netto, João ; Wagner, F. R. . Desenvolvimento de Sistemas Embarcados para Aplicações Críticas. In: *Escola Regional de Redes de Computadores, 2006, Passo Fundo*. Escola Regional de Redes de Computadores. Porto Alegre : Sociedade Brasileira de Computação, 2006. v. 1. p. 1-10.

Sitolino, Cláudio Luiz., Voz sobre IP – *Um estudo experimental 1999.* http://www.inf.ufrgs.br/pos/SemanaAcademica/Semana99/sitolino/sitolino.html. Acessado 16/08/2016.

Stallings, W. (2008). *Criptografia e segurança de redes: princípios e práticas.* PRENTICE HALL BRASIL.

Stapko, T. (2011). Practical Embedded Security: Building Secure Resource-Constrained Systems. *Embedded technology series.* Elsevier Science.

Thermos, P.; Takanen, A. *Securing VoIP networks: threats, vulnerabilities, countermeasures.* Boston: Pearson Education, 2007.

Walker, J. Q.; HICKS, J. T. *Taking charge of your VoIP project. [S.l.]:* Cisco Press, 2004.

APÊNDICE B – A Ser Submetido

An Approach to Performance Analysis in Embedded Devices Using Asterisk

Adauto Cavalcante Menezes*, Toniclay Andrade Nogueira†,
Edward David Moreno Ordonez‡ and Admilson de Ribamar Lima Ribeiro§
Computing Department
UFS - Segipe Federal University, Aracaju, Sergipe
E-mail: *adauto.cavalcant@gmail.com, †toniclay@globo.com, ‡edwdavid@gmail.com, §admilson@ufs.br

***Abstract*—Voice over IP (VoIP) communication will dominate the computing world for years to come. In order to perform VoIP communication, it is necessary to encode and decode the voice. This process consumes the main computational resources, as an example, we have the processor and memory. The telecommunication industries provide equipment with high values, which makes access to this technology still very restricted. Embedded devices are purposely constructed devices for certain applications, they execute systems with high criticality complex. Asterisk is a free software for voice over IP communication and has as main function to implement the functions of a telephone exchange. These technologies promise to reduce costs and maximize results. This work performs a performance analysis on three modern embedded devices (Raspberry Pi 3, Orange Pi Plus 2 and Banana Pi M3) using the Asterisk voice over IP communication system. The performance analysis consists in measuring the jitter, delay and bandwidth in each device with the protocols SIP and IAX2 with different CODEC'S.**

***Index Terms*—VoIP, Asterisk, Embedded Devices, Performance Analysis, Embedded Systems.**

I. Introduction

The term Voice over Internet Protocol (VoIP) is conceptualized as the voice communication in networks that use the Internet Protocol (IP), which developed with the appearance of IP Telephony, consists of the provision of telephony services using the network IP for the establishment of calls and voice communication [1]. In the middle of the year 1990, the definition of VoIP was consolidated, when the Internet Phone from VocalTec Communications, the first commercial software that enabled the communication of voice over IP, but with poor communication quality [2].

For Androulidakis [3], private telephone exchanges serve to communicate between internal telephones and communication with the public telephone network. There are IP communication systems or also called Private Branch Exchange (PBX) IP and Time Division Multiplexing (TDM) communication systems or conventional TDM PBXs. Their software can be offered proprietary or through open source. Communication with the external medium depends on the interface for connecting analog or digital lines, usually provided by telecommunication operators, and communication with internal telephones depends exclusively on the central office itself. Initially, the main advantage of the use of PBX communication systems was the cost of calling the internal lines, as there is an internal switching of circuits, which makes the call free.

Also according to [3], voice communication systems have gained popularity and now have functionalities, services that were not available from telecom operators such as hunt groups, call forwarding, and dial-by-extension. According to [4], the trend is to migrate to IP telephony.

In the year of 1999 came the software for communication of voice over IP Asterisk. According to [5], Asterisk is free software that performs all the functions of a conventional telephone exchange, developed by Digium. Currently, it receives numerous contributions from developers around the world, as it is a promising area of application that is constantly in development.

Thus, it is necessary to provide a mechanism to reduce expenses in the area of telephony, preferably with characteristics similar to that of a conventional telephone exchange, it is also necessary to expand and encourage the knowledge of students and researchers in the area of information technology and telecommunications. In this context, the importance of efficiency and cost reduction of the new generation of revolutionary personalized systems for voice over IP communication is justified. Thus, it is reinforced the idea that any advance of quality in this technology can propitiate a considerable increase in the quality of the voice communication, both in academic environment, as in the telecommunications industry, thus, we have the motivation that aims to accomplish this work.

From Asterisk, it is possible to implement a voice communication system on embedded, low-cost devices that provides the necessary mechanisms of a conventional telephone exchange. For this, we must perform an analysis of the behavior of IP phone calls on the devices through research jitter, delay and bandwidth. However, this work performs an approach to performance analysis on three embedded devices using Asterisk. At the end we can see which implementation has performed best to support the Asterisk voice over IP communication system. This paper is organized as follows, section II presents the related works, section III is dedicated to the methodology and

approach used. Section IV presents the experiments and results. Finally, section V presents the conclusion and future work.

II. Related Work

In this section, all works classified as significant and related to the present study are discussed. A systematic analysis was carried out in order to find relevant works in bases considered important in the area of computing (IEEE Xplore, Science Direct and the Brazilian Digital Library of Computation). The amount of work that Asterisk addresses in embedded devices is greatly reduced. In this way, work on performance analysis with the use of the SIP and IAX2 protocol was also researched, thus, it was possible to know the measurement techniques currently used.

In a paper entitled "Performance Analysis of VoIP Services over WiFi-based systems", the authors [6] presented an Alix hardware performance analysis for usage in VoIP systems under Wireless Fidelity (Wifi) networks, adopts a server with Embedded Asterisk software. This analysis consists of tests of the central processing unit (CPU) and RAM memory with a fixed number of simultaneous calls with the SIP and IAX2 protocols, and with the CODEC'S GSM, G.711 u / a, SPEEX and G.726 .

Still in [6], the authors could perform an energy efficiency analysis at the time of high number of simultaneous calls, as well as conduct simultaneous call behavior tests in an interconnection between Asterisk servers, this way making a more robust research.

In [7] present in their article a performance evaluation and Quality of Service (QOS) multimedia transmission (voice and video), using the SIP and IAX2 protocols based on an Asterisk server. The quality of service evaluated used some parameters of Qos, such as bandwidth, jitter and delay, in order to investigate the performance of different CODECS of voice and video.

In the work of [7], the authors could conduct concurrent call behavior tests with both softphones and servers. They could also design and develop an application for video transmission with support for the IAX2 protocol, in order to allow the completion of one of their tests listed in this work.

In the article "Implementation and Evaluation of Open Source Unified Communications for SMBs", the authors [8] presented the implementation and evaluation of a unified communication system composed of instant messaging and sharing (voice and video), voice messaging, VoIP communication and mobility. The evaluation covered only quantitative measurements of instantaneously supported telephone calls.

However, the authors [8] started their work with a unified communication system approach, nevertheless, in their experiments they performed only voice and video communication tests, thus, they did not demonstrate the efficiency of the system when used with several modules of the proposed unified communication system. Also in [8], the authors carried out a work with great contribution as for example the accomplishment of these same ones for the industry, researchers and academic community, however, there is still much to be done, experiments in embedded hardware devices, as well as the measurement of energy efficiency at times of high system consumption, are not addressed in this research.

The papers presented are of relevant content, they address an investigative practice for the concept of performance evaluation of a VoIP communication system that uses the SIP and IAX protocols.

Table I shows the hardwares, softwares, protocols and CODECS used in the experiments of the related works.

TABLE I
Data of Related Works

Authors	Hardware	Software	Protocol	Codec
[6]	PC X86 i7 8gb RAM Switch Iphone	AsteriskNow X-lite Zoiper Wireshark	SIP, IAX2	G.711 u/a Gsm G.722 Speex H.263 H.264 H.261 H.263p
[7]	PC x86 i7 3.40Ghz 8gb RAM	CentOS, Ubuntu, FreePBX, Sipp, Openfire	SIP	G.711 u/a G.729 Gsm
[8]	Alix 2d2	Voyage, Asterisk, Sipp	SIP, IAX2	G.711 u/a G.722 Gsm SpeeX G.726

It's worth noticing the use of various versions of the Linux operating system, as well as the use of Asterisk software in embedded device and in different hardware. In addition, all the works performed experiments in real environment, which makes the research more productive. The authors [8] and [6] addressed the use of the SIPP tool, which makes it possible to make several simultaneous calls. The SIP protocol, predominant in the three studies, was used more frequently, as well as CODEC G.711.

A good research work requires parameters for validating results. However, the authors did not address any metrics for validation, which leads to believe in the occurrence of possible errors in the results demonstrated.

III. Methodology and Approach

When defining a performance evaluation methodology, care must be taken for not to make common mistakes, such as lack of objectives, tendentious proposals, incorrect methods of evaluation, among others. To avoid such errors, the ideal is to adopt a systematic approach such as the one proposed by [9], which was applied in this work. To employ this methodology it becomes necessary to follow a sequence of steps.

The first step is the definition of objectives and the system. The second step is the preparation of the list of expected services and results. The third step is the selection of metrics, which establish the criteria for the comparison of performance. The fourth step is to compile the parameter list, in fact it is the list of parameters that affect performance. The fifth step, about the choice of factors for study, these factors are parameters that will suffer variations during the research. The sixth step is the selection of the evaluation technique, there are three techniques, which are simulation, analytical modeling and measurement. As a seventh step, you have the choice of the load, which consists of a list of service requests to the system. It is important to portray the actual use.

Then the eighth step is the planning of the experiments. As a ninth step, we have to analyze and interpret the data, in this step we must use adequate statistical techniques in order to consolidate the results obtained, in order to allow conclusions about the performance of the system. Already tenth and final step is the presentation of the results, in this step we must pay attention to the final presentation of the evaluation.

A. Application of the Methodology

In applying this methodology, we notice its importance, given the organized form that the work was conducted. Initially, it is necessary to define the objectives, then the scope of the system, the services offered, as well as the evaluation technique, which are shown below.

Goals:

- Perform a performance analysis on three low-cost embedded devices;
- Determine relevant factors in the performance of these equipments.

System:

- The system corresponds to a voice communication software over IP called Asterisk, it interacts with the medium through the reception and realization of telephone calls through IP Protocol.

Service:

- Voice over IP communication.

Assessment Technique:

- Measurement, therefore, is a useful technique for analyzing the performance of computer systems.

The activity was divided into five stages:

- Step 1 - Design and test scenario;
- Step 2 - Specification of metrics;
- Step 3 - Definition of parameters, factors and load;
- Step 4 - Planning and conducting the experiments;
- Step 5 - Statistical analysis of the results obtained.

The next subsection describes the first three steps, while the fourth and fifth are discussed in the Experiments and Results section.

B. Design and Test Scenario

To carry out the simulation, we must assume the existence of a computational model identical to the real production environment. In the literature, good descriptions for performance analysis in voice over IP communication systems have been found, for example, the work of [6] and [7]. The first work presents a performance analysis of the Alix 2D2 hardware for use in VoIP systems. The second one presents a performance assessment and Quality of Service (QOS) for multimedia transmission (voice and video).

Based on these works and technical specifications of commercially available equipment, the present work was carried out with three modern embedded devices and with the proposal of complementing the research already done, in this way, it effectively contributes to the telecommunications industry, for small and large companies, as well as, for the academic area, therefore, extends the realization of new researches.

This work verifies the possibility of verifying the use of modern embedded devices as servers of a voice over IP communication system called Asterisk using the SIP and IAX2 protocols and with the following CODEC'S: G.711a (Alaw), G.711u (Ulaw), Gsm, Speex, Ilbc and G.722, no other CODECs were used because of the limitations found in the work performed, such as free softphones used only to support these CODEC'S. The embedded devices discussed in this work have an average price of 50 dollars, thus, it is possible to consider as a solution of low cost.

The performance analysis consists in verifying the bandwidth, jitter and delay of the calls in each embedded device with the protocols and CODEC's mentioned above, for that, the tool was used for traffic analysis wireshark, software specialized in analysis of traffic in IP networks , widely used by researchers in the academic world, such as, for example, [7].

In order to perform this work, it was necessary to elaborate a test scenario, so the scenario included three modern and low cost embedded devices (Raspberry Pi 3, Banana Pi M3 and Orange Pi Plus 2), a RouterBoard 951g Mikrotik that made the switch , a Core i7 8GB RAM 500GB HD laptop. Fig.1 illustrates the architecture of the scenario designed to perform the experiments.

The RouterBoard Mikrotik that realizes the switch is responsible for interconnecting all the equipment in network. Each embedded device supports the Asterisk voice over IP communication system. The laptop has been allocated to perform the data collection with the Wireshark software.

Currently there is a great offer of embedded devices in the market, also called SBC (Single-Board Computers), with the most diverse configurations, for example, it is possible to mention Raspberry Pi Zero, Raspberry 2 Model B, Raspberry Pi 3 Model B, among others, Table II illustrates four models of last generation embedded devices, their configurations and price.

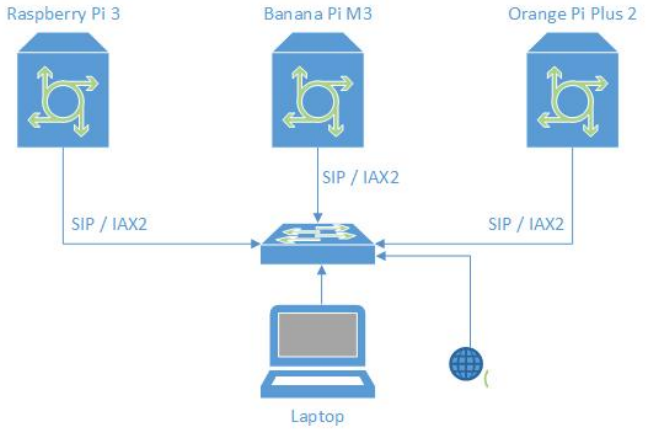


Fig. 1. Test scenario.

TABLE II
EMBEDDED DEVICES

Manufacturer	Raspberry Pi	Banana Pi	Orange Pi	Odroid
Model	3 Model B	M3	Plus 2	XU4
CPU Cores	4	8	4	4+4
CPU Freq.	1.2GHz	2GHz	1.5GHz	2.1GHz 1.5GHz
Memory	1GB DDR2	2GB DDR3	2GB DDR3	2GB DDR3
Storage	MicroSD	MicroSD USB Sata 2.0	MicroSD USB Sata 2.0	MicroSD eMMC
Linux	Yes	Yes	Yes	Yes
Price	$40	$59,99	$59,99	$59

According to [10], it is not easy to size a minimum hardware for the installation of the voice over IP communication system Asterisk, however, for this difficult question there is no precise answer. It is recommended to always use a hardware slightly beyond the needs, also suggests that if the system needs to support more than 20 simultaneous calls, a dedicated server should be used. Thus, for the accomplishment of this work, were chosen three devices of low cost and with good configurations, they are: Raspberry Pi 3, Banana Pi M3 and Orange Pi Plus 2.

IV. Experiments and Results

This section presents the implementation of the experiment, which consists of assembling the test scenario with the embedded devices individually, approaching the necessary software in the devices for the elaboration of the experiment, proceed with the process of collecting the jitter, delay and bandwidth with the Wireshark software, and finally, gauge the results. Fig.2 illustrates the actual testing scenario with the Banana Pi M3 device.

The collect was performed with the protocol SIP and IAX2, and with the Codec's supported by the free softphones addressed (Zoiper and X-lite). However, 33 collections (6 SIP and 5 IAX2 in each embedded device) were performed, each of calls lasting approximately 2 minutes, so it was possible to obtain a good quantitative of SIP and IAX2 packets to carry out the analysis, approximately 5,000 datagrams IP.

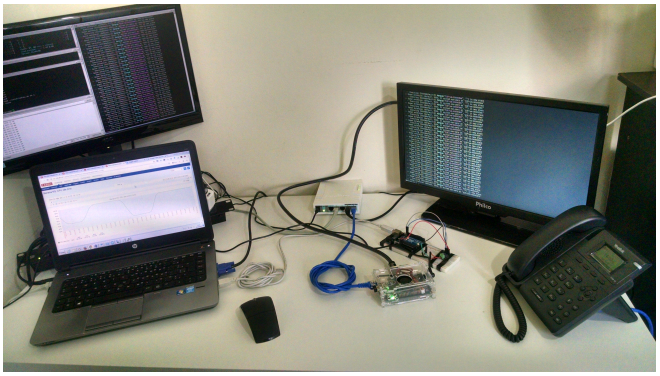


Fig. 2. Experiment running.

The Wireshark tool has filters that automatically perform jitter, bandwidth and delay analysis. Though, the mean was used as a parameter for the measurements. Notwithstanding, the mean bandwidth and delay were calculated manually by exporting the captured data to LibreOffice Calc, since Wireshark does not report the average, only the maximum value reached.

Thus, calls were made between the softphones and a self-service set up in the Asterisk of the embedded device. The tests were performed with the SIP and IAX2 protocols, as well as with the CODEC's mentioned above, in order to achieve the objective of this work with the measurements. The following subsections demonstrate the results collected with the SIP and IAX2 protocols on the embedded devices Raspberry Pi 3, Banana Pi M3 and Orange Pi Plus 2, with mean, standard deviation and confidence interval of 95%, as well as, holds brief discussions.

A. Bandwidth

Tables III and IV illustrate the bandwidth of the collected packets. We note that CODEC's G.711a, G.711u and G.722 have higher bandwidth rates and identical values, since CODEC's Gsm, Speex and Ilbc stand out because they have lower bandwidth, both with SIP protocol and with IAX2. The results found on the three embedded devices are similar. However, the distinction is clear in the approach of the SIP and IAX2 protocols, since it is observed that the IAX2 protocol consumes a smaller band in front of the SIP, as shown in Fig.3. It was not possible to collect the CODEC G.722 with the protocol IAX2 for not finding free softphone with CODEC support. Moreover, it is possible to state that the standard deviations and the confidence interval are minimal due to the high number of samples to carry out the approach.

In order to facilitate the identification of the devices in Fig.3, a code predecessor to the CODEC'S name has been

TABLE III
BANDWIDTH SIP

Raspberry Pi 3				
Codec's	**Average**	**Standard Deviation**	**Inferior Limit**	**Upper Limit**
G.711a	81,13	0,76	81,08	81,18
G.711u	81,11	0,77	81,06	81,15
Gsm	29,57	0,28	29,56	29,58
Speex	24,52	0,20	24,51	24,52
Ilbc	24,36	0,38	24,35	24,38
G.722	81,09	0,75	81,06	81,11
Banana Pi M3				
Codec's	**Average**	**Standard Deviation**	**Inferior Limit**	**Upper Limit**
G.711a	80,50	0,73	80,48	80,53
G.711u	80,48	0,72	80,46	80,5
Gsm	29,38	0,27	29,37	29,39
Speex	24,58	0,24	24,58	24,59
Ilbc	24,49	0,00	24,49	24,49
G.722	80,94	0,79	80,92	80,97
Orange Pi Plus 2				
Codec's	**Average**	**Standard Deviation**	**Inferior Limit**	**Upper Limit**
G.711a	80,51	0,74	80,49	80,53
G.711u	80,52	0,74	80,50	80,54
Gsm	29,40	0,27	29,39	29,41
Speex	24,58	0,23	24,57	24,58
Ilbc	24,49	0,00	24,49	24,49
G.722	81,06	0,76	81,04	81,08

TABLE IV
BANDWIDTH IAX2

Raspberry Pi 3				
Codec's	**Average**	**Standard Deviation**	**Inferior Limit**	**Upper Limit**
G.711a	74,21	0,72	74,19	74,23
G.711u	74,20	0,72	74,18	74,22
Gsm	22,99	0,22	22,98	23,00
Speex	18,14	0,17	18,13	18,14
Ilbc	20,13	0,02	20,13	20,13
Banana Pi M3				
Codec's	**Average**	**Standard Deviation**	**Inferior Limit**	**Upper Limit**
G.711a	74,37	0,78	74,35	74,4
G.711u	74,33	0,78	74,31	74,36
Gsm	23,03	0,23	23,03	23,04
Speex	18,18	0,18	18,17	18,18
Ilbc	20,03	0,32	20,02	20,04
Orange Pi Plus 2				
Codec's	**Average**	**Standard Deviation**	**Inferior Limit**	**Upper Limit**
G.711a	74,57	0,70	74,55	74,60
G.711u	74,25	0,73	74,23	74,28
Gsm	23,01	0,23	23,00	23,02
Speex	18,16	0,18	18,15	18,16
Ilbc	20,13	0,19	20,12	20,13

inserted, so that, (R.) for the Raspberry Pi device 3, (B.) for the Banana Pi device M3 and (O.) , respectively for the Orange Pi Plus 2 device.

B. Delay

Tables V and VI demonstrate the delay of the collected packets. It is possible to observe that Delay with the SIP protocol is in accordance with the standard [12], which

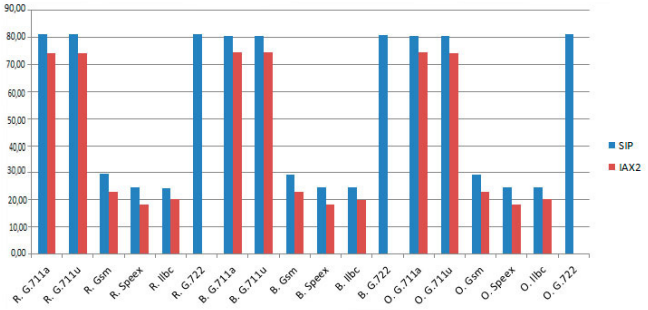


Fig. 3. Bandwidth SIP e IAX2.

TABLE V
DELAY SIP

Raspberry Pi 3				
Codec's	**Average**	**Standard Deviation**	**Inferior Limit**	**Upper Limit**
G.711a	20,02	0,30	20,00	20,04
G.711u	20,02	0,30	20,00	20,04
Gsm	20,01	0,12	20,01	20,02
Speex	20,02	0,12	20,01	20,03
Ilbc	29,92	0,25	29,91	29,93
G.722	20,00	0,30	19,99	20,01
Banana Pi M3				
Codec's	**Average**	**Standard Deviation**	**Inferior Limit**	**Upper Limit**
G.711a	20,00	0,06	20,00	20,01
G.711u	20,00	0,05	20,00	20,01
Gsm	20,00	0,06	20,00	20,01
Speex	20,00	0,12	20,00	20,01
Ilbc	30,01	0,41	29,99	30,02
G.722	20,00	0,27	20,00	20,01
Orange Pi Plus 2				
Codec's	**Average**	**Standard Deviation**	**Inferior Limit**	**Upper Limit**
G.711a	20,00	0,06	20,00	20,01
G.711u	20,00	0,07	20,00	20,01
Gsm	20,00	0,11	20,00	20,01
Speex	20,00	0,12	20,00	20,01
Ilbc	30,01	0,17	30,00	30,01
G.722	20,00	0,10	20,00	20,01

establishes that the default delay of the RTP packet should be 20ms. However, there is an exception to the CODEC Ilbc, which stands out with a value of 50% higher than the others, as shown in Fig.4. This is due to its coding algorithm, which performs high compression of the audio and consequently increases the transmission delay.

Delay with the IAX2 protocol tends to zero with all CODEC's addressed in this research, which validates its proposal as a new protocol standard. According to [11] it is an open protocol that carries the transport of signaling and the media. In addition, the IAX2 protocol also proposes to eliminate any transmission delays. It is also observed that the standard deviations and the confidence interval are minimal due to the high number of samples to carry out the approach.

TABLE VI
DELAY IAX2

Raspberry Pi 3				
Codec's	**Average**	**Standard Deviation**	**Low Limit**	**Upper Limit**
G.711a	0,02	0,00	0,02	0,02
G.711u	0,02	0,00	0,02	0,02
Gsm	0,01	0,02	0,02	0,02
Speex	0,02	0,00	0,02	0,02
Ilbc	0,03	0,00	0,03	0,03
Banana Pi M3				
Codec's	**Average**	**Standard Deviation**	**Low Limit**	**Upper Limit**
G.711a	0,02	0,01	0,02	0,02
G.711u	0,02	0,01	0,02	0,02
Gsm	0,02	0,01	0,02	0,02
Speex	0,02	0,01	0,02	0,02
Ilbc	0,03	0,01	0,03	0,03
Orange Pi Plus 2				
Codec's	**Average**	**Standard Deviation**	**Low Limit**	**Upper Limit**
G.711a	0,02	0,00	0,02	0,02
G.711u	0,02	0,00	0,02	0,02
Gsm	0,02	0,00	0,02	0,02
Speex	0,02	0,00	0,02	0,02
Ilbc	0,03	0,00	0,03	0,03

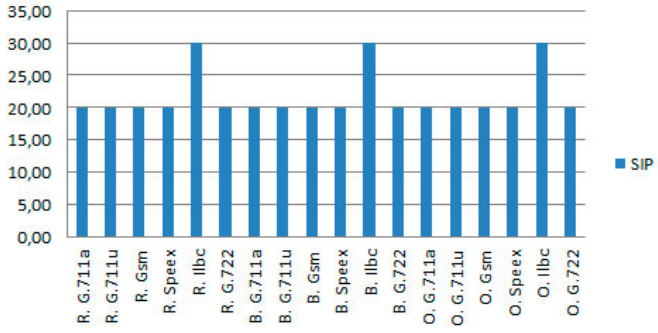


Fig. 4. Delay SIP.

C. Jitter

Tables VII and VIII illustrate the jitter of the collected packets. It is possible to observe that, with both the SIP protocol and the IAX2 protocol, jitter tends to zero, this is because both protocols have as strategy to keep the frames in a buffer, in order to allow the slower frames arrive in time to be played in the correct sequence. The higher the amount of jitter, the greater the number of frames in the buffer in order to minimize jitter in VoIP calls.

TABLE VII
JITTER SIP

Raspberry Pi 3				
Codec's	**Average**	**Standard Deviation**	**Low Limit**	**Upper Limit**
G.711a	0,15	0,09	0,15	0,15
G.711u	0,15	0,09	0,15	0,15
Gsm	0,10	0,23	0,09	0,10
Speex	0,21	0,15	0,21	0,21
Ilbc	0,24	0,15	0,23	0,25
G.722	0,15	0,10	0,14	0,15
Banana Pi M3				
Codec's	**Average**	**Standard Deviation**	**Low Limit**	**Upper Limit**
G.711a	0,03	0,01	0,03	0,03
G.711u	0,03	0,01	0,03	0,03
Gsm	0,03	0,01	0,03	0,03
Speex	0,06	0,02	0,06	0,06
Ilbc	0,24	0,08	0,24	0,25
G.722	0,15	0,09	0,14	0,15
Orange Pi Plus 2				
Codec's	**Average**	**Standard Deviation**	**Low Limit**	**Upper Limit**
G.711a	0,04	0,01	0,04	0,04
G.711u	0,04	0,01	0,04	0,04
Gsm	0,06	0,03	0,06	0,06
Speex	0,06	0,02	0,06	0,06
Ilbc	0,09	0,04	0,10	0,10
G.722	0,06	0,03	0,06	0,06

TABLE VIII
JITTER IAX2

Raspberry Pi 3				
Codec's	**Average**	**Standard Deviation**	**Low Limit**	**Upper Limit**
G.711a	0,02	0,19	0,01	0,02
G.711u	0,02	0,19	0,01	0,02
Gsm	0,00	0,00	0,00	0,00
Speex	0,02	0,19	0,01	0,02
Ilbc	0,02	0,19	0,01	0,03
Banana Pi M3				
Codec's	**Average**	**Standard Deviation**	**Low Limit**	**Upper Limit**
G.711a	0,02	0,19	0,01	0,03
G.711u	0,02	0,19	0,01	0,03
Gsm	0,02	0,19	0,01	0,02
Speex	0,02	0,19	0,01	0,02
Ilbc	0,03	0,19	0,01	0,02
Orange Pi Plus 2				
Codec's	**Average**	**Standard Deviation**	**Low Limit**	**Upper Limit**
G.711a	0,02	0,19	0,01	0,02
G.711u	0,02	0,19	0,01	0,02
Gsm	0,02	0,19	0,01	0,02
Speex	0,02	0,19	0,01	0,02
Ilbc	0,02	0,19	0,01	0,02

V. Conclusion and Future Work

In this article, a performance analysis approach was performed on three state-of-the-art embedded devices using the Asterisk voice over IP communication system, so it was possible to measure jitter, delay and flow with the SIP and IAX2 protocols with the CODEC's G.711a (Alaw), G.711u (Ulaw), Gsm, Ilbc, Speex and G.722.

Measurements were made to compare the three embedded devices using Asterisk. However, the results showed great similarity in the data, both with the SIP protocol and with the IAX2 protocol, this in the network requirement. As a continuation of this research, we intend to evaluate the maximum amount of simultaneous calls for each embedded device and in parallel, to monitor the memory, processing and energy consumption, both in normal calls and in transcoded calls with the same protocols addressed in this work.

As future works, it is proposed to perform these experiments with the Codec Opus, since it was not possible to perform the compilation due to incompatibility of the embedded device and limitations of the softphones addressed. It is possible also perform the same approach with the IAX2 protocol using encryption.

References

[1] P. S. M. Bernal, *Voz sobre protocolo IP: A nova realidade da telefonia*, São Paulo, Brazil: Erica, 2007.

[2] S. Colcher, A. T. A. Gomes, A. O. Silva, G. L. S. Filho and L. F. G. Soares, *VoIP: voz sobre IP*, Rio de Janeiro, Brazil: Elsevier, 2005.

[3] I. I. Androulidakis, *Mobile Phone Security and Forensics: A Practical Approach*, Springer, 2016.

[4] A. Sulkin, *PBX systems for IP telephony*, New York, NY, United States, McGraw-Hill, 2002.

[5] R. Bryant, L. Madsen and J. V. Meggelen *Asterisk: The definitive guide*, 4 ed. United States: O'Reilly Media, 2013.

[6] D. Villacis, A. R. Freddy and L. A. R. Cueva *Performance analysis of VoIP Services over WiFi-based systems*, In: Communications and Computing (COLCOM) IEEE Colombian Conference on. IEEE, 2013. p. 1-6.

[7] N. M. Edan, S. Turner, A. Al-Sherbaz and S. Ajit *Performance evaluation of QoS using SIP and IAX2 VVoIP protocols with CODECS*, In: Communications and Computing (COLCOM) IEEE Colombian Conference on. IEEE, 2013. p. 1-6.

[8] A. D. Tesfamicael, V. Liu, W. Caelli and J. Zureo *Implementation and evaluation of open source unified communications for SMBs*, In: Computational Intelligence and Communication Networks (CICN), 2014 International Conference on. IEEE, 2014. p. 1243-1248.

[9] R. Jain *The Art of Computer Systems Performance Analysis: techniques for experimental design, measurement, simulation, and modeling*, John Wiley and Sons, 1991.

[10] Digium *Dimensioning*, (2017) Available: http://www.digium.com/blog/2012/09/25/asterisk-dimensioning-what-server-do-i-need

[11] RFC5456 *Inter-Asterisk eXchange Version 2*, (2017) Available: https://tools.ietf.org/html/rfc5456

[12] RFC1890 *RTP Profile for Audio and Video Conferences with Minimal Control*, (2017) Available: https://tools.ietf.org/html/rfc1890

APÊNDICE C – A Ser Submetido

Performance and energy efficiency analysis of embedded devices using SIP and IAX2 protocols with Asterisk

Adauto Cavalcante Menezes*, Toniclay Andrade Nogueira†,
Edward David Moreno Ordonez‡ and Admilson de Ribamar Lima Ribeiro§
Computing Department
UFS - Segipe Federal University, Aracaju, Sergipe
E-mail: *adauto.cavalcant@gmail.com, †toniclay@globo.com, ‡edwdavid@gmail.com, §admilson@ufs.br

***Abstract*—Voice over IP (VoIP) communication will dominate the computing world for years to come. In order to perform VoIP communication, it is necessary to encode and decode the voice. This process consumes the main computational resources, as an example, we have the processor and memory. The telecommunication industries provide equipment with high values, which makes access to this technology still very restricted. Embedded devices are purposely constructed devices for certain applications, they execute systems with high criticality complex. Asterisk is a free software for voice over IP communication and has as main function to implement the functions of a telephone exchange. These technologies promise to reduce costs and maximize results. This work performs a performance analysis on three modern embedded devices (Raspberry Pi 3, Orange Pi Plus 2 and Banana Pi M3) using the Asterisk voice over IP communication system. Performance analysis consists of measuring the number of concurrent calls supported on each device with the SIP and IAX2 protocols with different CODEC's and in parallel monitoring the RAM, processing and power consumption.**

***Index Terms*—VoIP, Asterisk, Embedded Devices, Performance Analysis, Embedded Systems.**

I. Introduction

The term Voice over Internet Protocol (VoIP) is conceptualized as the voice communication in networks that use the Internet Protocol (IP), which developed with the appearance of IP Telephony, consists of the provision of telephony services using the network IP for the establishment of calls and voice communication [1]. In the middle of the year 1990, the definition of VoIP was consolidated, when the Internet Phone from VocalTec Communications, the first commercial software that enabled the communication of voice over IP, but with poor communication quality [2].

For Androulidakis [3], private telephone exchanges serve to communicate between internal telephones and communication with the public telephone network. There are IP communication systems or also called Private Branch Exchange (PBX) IP and Time Division Multiplexing (TDM) communication systems or conventional TDM PBXs. Their software can be offered proprietary or through open source. Communication with the external medium depends on the interface for connecting analog or digital lines, usually provided by telecommunication operators, and communication with internal telephones depends exclusively on the central office itself. Initially, the main advantage of the use of PBX communication systems was the cost of calling the internal lines, as there is an internal switching of circuits, which makes the call free.

Also according to [3], voice communication systems have gained popularity and now have functionalities, services that were not available from telecom operators such as hunt groups, call forwarding, and dial-by-extension. According to [4], the trend is to migrate to IP telephony.

In the year of 1999 came the software for communication of voice over IP Asterisk. According to [5], Asterisk is free software that performs all the functions of a conventional telephone exchange, developed by Digium. Currently, it receives numerous contributions from developers around the world, as it is a promising area of application that is constantly in development.

The telecommunication industries provide equipment with high values and mostly proprietary equipment. This makes it difficult to access VoIP technology.

Thus, it is necessary to provide a mechanism to reduce expenses in the area of telephony, preferably with characteristics similar to that of a conventional telephone exchange, it is also necessary to expand and encourage the knowledge of students and researchers in the area of information technology and telecommunications. In this context, the importance of efficiency and cost reduction of the new generation of revolutionary personalized systems for voice over IP communication is justified. Thus, it is reinforced the idea that any advance of quality in this technology can propitiate a considerable increase in the quality of the voice communication, both in academic environment, as in the telecommunications industry, thus, we have the motivation that aims to accomplish this work.

From Asterisk, it is possible to implement a voice communication system on embedded, low-cost devices that provides the necessary mechanisms of a conventional telephone exchange. For this, a performance analysis

should be performed on three modern embedded devices (Raspberry Pi 3, Orange Pi Plus 2 and Banana Pi M3), using the Asterisk voice over IP communication system. Performance analysis consists of measuring the number of concurrent calls supported on each device with the SIP and IAX2 protocols with different CODEC's and in parallel monitoring the RAM, processing and power consumption. At the end we can see which implementation has performed best to support the Asterisk voice over IP communication system. This paper is organized as follows, section II presents the related works, section III is dedicated to the methodology and approach used. Section IV presents the experiments and results. Finally, section V presents the conclusion and future work.

II. Related Work

In this section, all works classified as significant and related to the present study are discussed. A systematic analysis was carried out in order to find relevant works in bases considered important in the area of computing (IEEE Xplore, Science Direct and the Brazilian Digital Library of Computation). The amount of work that Asterisk addresses in embedded devices is greatly reduced. In this way, work on performance analysis with the use of the SIP and IAX2 protocol was also researched, thus, it was possible to know the measurement techniques currently used.

In a paper entitled "Performance Analysis of VoIP Services over WiFi-based systems", the authors [6] presented an Alix hardware performance analysis for usage in VoIP systems under Wireless Fidelity (Wifi) networks, adopts a server with Embedded Asterisk software. This analysis consists of tests of the central processing unit (CPU) and RAM memory with a fixed number of simultaneous calls with the SIP and IAX2 protocols, and with the CODEC'S GSM, G.711 u / a, SPEEX and G.726 . Still in [6], the authors could perform an energy efficiency analysis at the time of high number of simultaneous calls, as well as conduct simultaneous call behavior tests in an interconnection between Asterisk servers, this way making a more robust research.

In [7] present in their article a performance evaluation and Quality of Service (QOS) multimedia transmission (voice and video), using the SIP and IAX2 protocols based on an Asterisk server. The quality of service evaluated used some parameters of Qos, such as bandwidth, jitter and delay, in order to investigate the performance of different CODECS of voice and video. In the work of [7], the authors could conduct concurrent call behavior tests with both softphones and servers. They could also design and develop an application for video transmission with support for the IAX2 protocol, in order to allow the completion of one of their tests listed in this work.

In the article "Implementation and Evaluation of Open Source Unified Communications for SMBs", the authors [8] presented the implementation and evaluation of a unified communication system composed of instant messaging and sharing (voice and video), voice messaging, VoIP communication and mobility. The evaluation covered only quantitative measurements of instantaneously supported telephone calls. However, the authors [8] started their work with a unified communication system approach, nevertheless, in their experiments they performed only voice and video communication tests, thus, they did not demonstrate the efficiency of the system when used with several modules of the proposed unified communication system. Also in [8], the authors carried out a work with great contribution as for example the accomplishment of these same ones for the industry, researchers and academic community, however, there is still much to be done, experiments in embedded hardware devices, as well as the measurement of energy efficiency at times of high system consumption, are not addressed in this research.

In [9], their article "Embedded Implementation of an IP-PBX / VoIP Gateway", propose the idea of designing and implementing an embedded PBX-IP gateway, which uses low cost and open source solutions. The system integrates FPGA hardware and incorporates software into external memory. Their experiment consists of an ML501 FPGA hardware, with embedded asterisk software and two pcx86 computers, one connected to serial port and the other connected to the ethernet network. The authors [9], have performed only a test of Asterisk software support in the hardware embedded FPGA. The authors performed only one network connectivity test. However, they did not perform tests to gauge the number of concurrent calls supported, measurement of power consumption, processing and memory.

The papers presented above are of relevant content, they address an investigative practice for the concept of performance evaluation of a VoIP communication system that uses the SIP and IAX protocols.

Table I shows the hardwares, softwares, protocols and CODECS used in the experiments of the related works, as well as in our work. Table II shows a comparison between the activities carried out in our work and those presented in the related works. It is observed that our is more complete. In the following sections, the implementation details of performance analysis and energy efficiency in embedded devices using Asterisk, as well as prototyping and the measurements taken to extract the results will be described.

It's worth noticing the use of various versions of the Linux operating system, as well as the use of Asterisk software in embedded device and in different hardware. In addition, all the works performed experiments in real environment, which makes the research more productive. The authors [8] and [6] addressed the use of the SIPP tool, which makes it possible to make several simultaneous calls. The SIP protocol, predominant in the three studies, was used more frequently, as well as CODEC G.711.

A good research work requires parameters for validat-

TABLE I
Data of Related Works

Authors	Hardware	Software	Protocol	Codec
[6]	PC X86 i7 8gb RAM Switch Iphone	AsteriskNow X-lite Zoiper Wireshark	SIP IAX2	G.711 u G711a Gsm G.722 Speex H.263 H.264 H.261 H.263p
[7]	PC x86 i7 3.40Ghz 8GB RAM	CentOS Ubuntu FreePBX Sipp Openfire	SIP	G.711 u G.711a G.729 Gsm
[8]	Alix 2d2	Voyage Asterisk Sipp	SIP IAX2	G.711 u G.711a G.722 Gsm Speex G.726
[9]	ML501 FPGA	Linux Asterisk	-	-
This work	Raspberry Pi 3, Banana Pi M3, Orange Pi Plus 2, Mikrotik 951g.	Raspbian Armbian Wireshark Zabbix Zoiper X-lite	SIP IAX2	G.711a G.711u G.722 Gsm Ilbc Speex

TABLE II
Activities carried

Qualities	[6]	[7]	[8]	[9]	This work
Embedded device	x	-	-	x	x
Simultaneous calls sip	x	-	x	-	x
Simultaneous calls iax2	x	-	-	-	x
Memory Consumption	x	-	-	-	x
CPU consumption	x	-	x	-	x
Energy consumption	-	-	-	-	x
Validation metric	-	-	-	-	x

ing results. However, However, all the above mencioned authors did not address any metrics for validation, which leads to believe in the occurrence of possible errors in the results demonstrated.

III. Methodology and Approach

When defining a performance evaluation methodology, care must be taken for not to make common mistakes, such as lack of objectives, tendentious proposals, incorrect methods of evaluation, among others. To avoid such errors, the ideal is to adopt a systematic approach such as the one proposed by [10], which was applied in this work. To employ this methodology it becomes necessary to follow a sequence of steps.

The first step is the definition of objectives and the system. The second step is the preparation of the list of expected services and results. The third step is the selection of metrics, which establish the criteria for the comparison of performance. The fourth step is to compile the parameter list, in fact it is the list of parameters that affect performance. The fifth step, about the choice of factors for study, these factors are parameters that will suffer variations during the research. The sixth step is the selection of the evaluation technique, there are three techniques, which are simulation, analytical modeling and measurement. As a seventh step, you have the choice of the load, which consists of a list of service requests to the system. It is important to portray the actual use.

Then the eighth step is the planning of the experiments. As a ninth step, we have to analyze and interpret the data, in this step we must use adequate statistical techniques in order to consolidate the results obtained, in order to allow conclusions about the performance of the system. Already tenth and final step is the presentation of the results, in this step we must pay attention to the final presentation of the evaluation.

A. Application of the Methodology

In applying this methodology, we notice its importance, given the organized form that the work was conducted. Initially, it is necessary to define the objectives, then the scope of the system, the services offered, as well as the evaluation technique, which are shown below.

Goals:

- Perform a performance analysis on three low-cost embedded devices;
- Determine relevant factors in the performance of these equipments.

System:

- The system corresponds to a voice communication software over IP called Asterisk, it interacts with the medium through the reception and realization of telephone calls through IP Protocol.

Service:

- Voice over IP communication.

Assessment Technique:

- Measurement, therefore, is a useful technique for analyzing the performance of computer systems.

The activity was divided into five stages:

- Step 1 - Design and test scenario;
- Step 2 - Specification of metrics;
- Step 3 - Definition of parameters, factors and load;
- Step 4 - Planning and conducting the experiments;
- Step 5 - Statistical analysis of the results obtained.

The next subsection describes the first three steps, while the fourth and fifth are discussed in the Experiments and Results section.

B. Design and Test Scenario

To carry out the prototyping, we must assume the existence of a computational model identical to the real production environment. In the literature, good descriptions for performance analysis in voice over IP communication

systems have been found, for example, the work of [6] and [7]. The first work presents a performance analysis of the Alix 2D2 hardware for use in VoIP systems. The second one presents a performance assessment and Quality of Service (QOS) for multimedia transmission (voice and video).

Based on these works and technical specifications of commercially available equipment, the present work was carried out with three modern embedded devices and with the proposal of complementing the research already done, in this way, it effectively contributes to the telecommunications industry, for small and large companies, as well as, for the academic area, therefore, extends the realization of new researches.

This work verifies the possibility of verifying the use of modern embedded devices as servers of a voice over IP communication system called Asterisk using the SIP and IAX2 protocols and with the following CODEC'S: G.711a (Alaw), G.711u (Ulaw), Gsm, Speex, Ilbc and G.722, no other CODECs were used because of the limitations found in the work performed, such as free softphones used only to support these CODEC'S. The embedded devices discussed in this work have an average price of 50 dollars, thus, it is possible to consider as a solution of low cost.

The performance analysis consists in verifying the quantitative simultaneous calls supported in each embedded device with the protocols and CODEC's mentioned above and in parallel, to carry out the monitoring of the consumption of RAM, processing and energy. For this, the tool was used for traffic analysis wireshark, software specialized in traffic analysis in IP networks, widely used by researchers in the academic environment, for example, [7].

In order to perform this work, it was necessary to elaborate a test scenario, so the scenario included three modern and low cost embedded devices (Raspberry Pi 3, Banana Pi M3 and Orange Pi Plus 2), a RouterBoard 951g Mikrotik that made the switch , a Core i7 8GB RAM 500GB HD laptop. Fig.1 illustrates the architecture of the scenario designed to perform the experiments.

The RouterBoard Mikrotik that realizes the switch is responsible for interconnecting all the equipment in network. Each embedded device supports the Asterisk voice over IP communication system. The laptop has been allocated to perform the data collection with the Wireshark software.

Currently there is a great offer of embedded devices in the market, also called SBC (Single-Board Computers), with the most diverse configurations, for example, it is possible to mention Raspberry Pi Zero, Raspberry 2 Model B, Raspberry Pi 3 Model B, among others. Table III illustrates four models of last generation embedded devices, their configurations and price.

According to [11], it is not easy to size a minimum hardware for the installation of the voice over IP communication system Asterisk, however, for this difficult question there is no precise answer. It is recommended to always use a hardware slightly beyond the needs, also suggests that

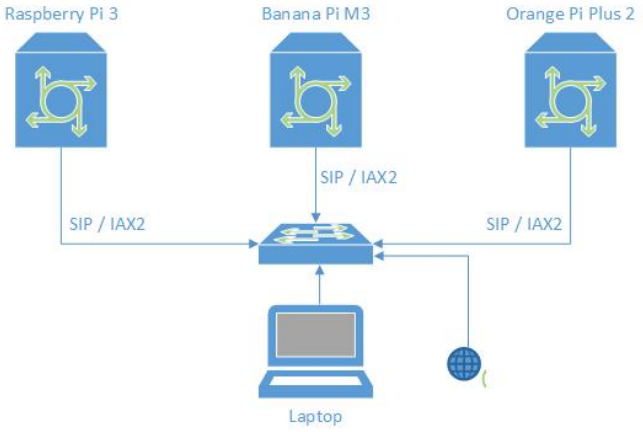


Fig. 1. Test scenario.

TABLE III
EMBEDDED DEVICES

Manufacturer	Raspberry Pi	Banana Pi	Orange Pi	Odroid
Model	3 Model B	M3	Plus 2	XU4
CPU Cores	4	8	4	4+4
CPU Freq.	1.2GHz	2GHz	1.5GHz	2.1GHz 1.5GHz
Memory	1GB DDR2	2GB DDR3	2GB DDR3	2GB DDR3
Storage	MicroSD	MicroSD USB Sata 2.0	MicroSD USB Sata 2.0	MicroSD eMMC
Linux	Yes	Yes	Yes	Yes
Price	$40	$59,99	$59,99	$59

if the system needs to support more than 20 simultaneous calls, a dedicated server should be used. Thus, for the accomplishment of this work, were chosen three devices of low cost and with good configurations, they are: Raspberry Pi 3, Banana Pi M3 and Orange Pi Plus 2.

IV. EXPERIMENTS AND RESULTS

This section presents the implementation of the experiment, which consists of: assembling the test scenario with the embedded devices individually; accomplish software approach in the devices for the elaboration of the experiment; implement a code in Shell Script in order to generate loads of SIP and IAX calls with CODEC?s G.711u (Ulaw), G.711a (Alaw), Gsm, Speex, Ilbc and G.722; run high call flows and in parallel carry out the monitoring of the consumption of processing, RAM and energy; and finally, to gauge the results. Fig.2 illustrates the actual testing scenario with the Banana Pi M3 device.

The tests were performed in order to reach the maximum number of calls supported with no occurrence of errors. Thus, several attempts were made to until you find the expected result. Thereafter, 10 replicates were performed to validate the occurrence of no error. For it is

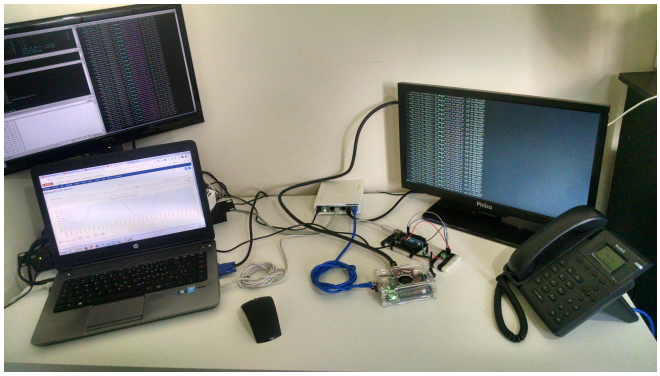


Fig. 2. Experiment running.

a static result, there is no variation of the mean and thus a greater number of repetitions can be dispensed with.

In order to carry out the generation of simultaneous SIP call load, Asterisk was used in a virtual machine installed on the laptop, and a dialer was implemented in Shell Script. For calls with the IAX2 protocol, it was held the exchange of the channel on the dialer, from SIP to IAX2. The dialer code is shown in Fig.3.

```
#!/bin/bash
aux1=$1;
aux2=$2;
CONTADOR=0
CONTADOR2=0
echo "Enviando $1 chamadas $2 vezes"

while [ $CONTADOR2 -lt $aux2 ]; do
echo "$CONTADOR2 x"
while [ $CONTADOR -lt $aux1 ]; do
echo "Channel: SIP/embarcado/5000" >>
/etc/asterisk/callfiles/teste$CONTADOR_$CONTADOR2.call
echo "CallerID: embarcado" >>
 /etc/asterisk/callfiles/teste$CONTADOR_$CONTADOR2.call
echo "MaxRetries: 0" >>
/etc/asterisk/callfiles/teste$CONTADOR_$CONTADOR2.call
echo "RetryTime: 60" >>
 /etc/asterisk/callfiles/teste$CONTADOR_$CONTADOR2.call
echo "WaitTime: 30" >>
/etc/asterisk/callfiles/teste$CONTADOR_$CONTADOR2.call
echo "Context: default" >>
/etc/asterisk/callfiles/teste$CONTADOR_$CONTADOR2.call
echo "Extension: 3000" >>
/etc/asterisk/callfiles/teste$CONTADOR_$CONTADOR2.call

mv /etc/asterisk/callfiles/teste$CONTADOR_$CONTADOR2.call
/var/spool/asterisk/outgoing/
let CONTADOR=CONTADOR+1;
done
sleep 10s;
CONTADOR=0
let CONTADOR2=CONTADOR2+1;
done
```

Fig. 3. Dialer code.

Thus, two files were generated to perform high numbers called simultaneous, called CallFilePlay.sh and CallFileCodec.sh. The first for normal calls, the second for calls with transcoding. The only difference between the two files is that in CallFileCodec.sh the extension number on the dialer is different. In this way, these files were inserted in the Asterisk / etc / Asterisk directory on the dialer virtual machine. And so, it was possible to start the tests.

It was also necessary to configure an audio file called test.gsm, with 3 minutes, and include it in Asterisk's "sounds / var / lib / asterisk / sounds" directory, whose purpose is for Asterisk to receive calls from the load generator, forward them to self-service and then play the audio. In this way, occurs the media flow limited time, enough to run the concurrent call test.

To start the tests, the virtual machine was used on the laptop to run the dialer load generator, and in parallel the monitoring of the consumption of RAM memory and processing with Zabbix software; monitoring the energy consumption with the circuit INA219, as well as verification of errors occurring in the processing of calls, through of the debug in Asterisk.

A Yalink T19p E2 IP telephone was also used in order to validate the quality of the call at high load consumption. However, it was not possible to validate the quality with the IAX2 protocol and neither with the Speex and Gsm CODEC's with the SIP protocol, due to the license limitations of the telephone set.

As the tests were carried out, the CODECs were modified, as well as the protocol. The experiments were divided into 4 stages, which will be described of in next subsections, as well as their results.

A. SIP calls

Table IV illustrates the data collected using the SIP protocol. It is possible to observe that the GSM CODEC supported the largest number of calls in the Raspberry Pi 3 and Banana Pi M3 devices. However, there is a high consumption of RAM. Nevertheless, given the RAM capacity of the Raspberry Pi 3 device of 1GB RAM and Banana Pi M3 of 2GB, this result becomes insignificant. Though, the GSM CODEC has the lowest processing. The CODEC G.711a stands out because it is the CODEC that consumes less energy and supports a significant number of calls because it is an embedded device.

TABLE IV
DATA COLLECTED USING THE SIP PROTOCOL.

Raspberry Pi 3 SIP							
CODEC's	**Repose**	**Alaw**	**Ulaw**	**Gsm**	**Ilbc**	**Speex**	**G.722**
Simultaneous calls	0	257	251	291	211	211	241
Memory (MB)	48,48	95,01	93,92	121,13	102	94,66	125,75
Processing (%)	0,05	64,36	70,55	55,72	100	100	78,89
Current (energy)	390,76	486,91	691,31	644,37	838,33	848,5	726,43
Banana Pi M3 SIP							
Simultaneous calls	0	261	251	305	229	221	249
Memory (MB)	89,04	149,62	139,13	180,09	142,13	142,68	145,57
Processing (%)	0,51	69,05	72,43	63,89	100	100	73,16
Current (energy)	686,11	890,39	950,45	934,21	1038,25	988,64	905
Orange Pi Plus 2 SIP							
Simultaneous calls	0	247	261	251	241	241	251
Memory (MB)	81,97	113,77	125,5	113,51	122,32	124,46	117,83
Processing (%)	0,56	98,97	99,82	99,48	98,67	98,53	99,34
Current (energy)	805,33	1133,69	1138,51	1119,26	1130,21	1135,64	1137,43

On the other hand the Orange Pi Plus 2 device, got high processing in all tests. Given this information, it is possible to state that this device is not ideal for use with Asterisk.

Fig.4 illustrates the behavior of the calls made on the 3 embedded devices, which allows better evaluation and

comparison of the data. However, we note that the Orange Pi Plus 2 device has identical results with all CODEC's addressed in this research, which reinforces the thesis that this device is not adequate to use with the software Asterisk.

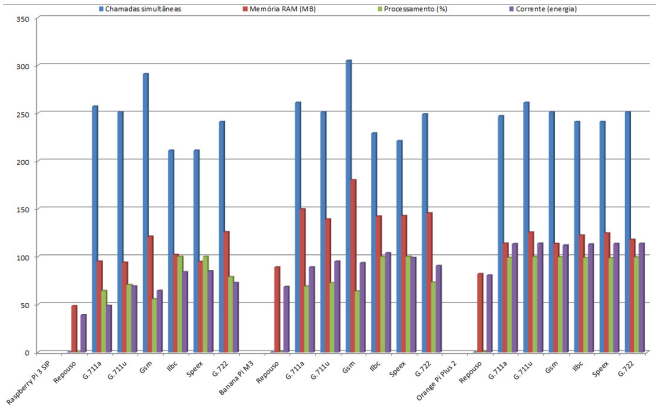


Fig. 4. SIP comparative analysis.

B. SIP calls with Transcoding

Table V shows the results collected on calls made with the SIP protocol and with transcoding. We note that with transcoding, the performance of the devices is greatly reduced. Even so, processing of the Orange Pi Plus 2 device remains high. This, once again, reinforces the claim that such a device is not suitable for use Asterisk. In the other devices, it is observed that the G.711a and G.711u CODECs have a higher number of concurrent calls supported and lower processing, memory and power consumption. In addition, the Raspberry Pi 3 device stands out with a considerable consumption of processing with CODEC's G.711a and G.711u, as shown in Fig.5.

TABLE V
SIP WITH TRANSCODING.

Raspberry Pi 3							
CODEC's	**Repouso**	**Alaw**	**Ulaw**	**Gsm**	**Ilbc**	**Speex**	**G.722**
Simultaneous calls	0	119	121	111	87	101	91
Memory (MB)	48,48	87,17	95,98	99,47	148,3	247,16	97,49
Processing (%)	0,05	3,97	41,87	68,83	100	100	93,7
Current (energy)	390,76	438,85	592,64	592,04	821,49	577,56	654,27
Banana Pi M3							
Simultaneous calls	0	131	129	131	109	111	129
Memory (MB)	89,04	136,14	115,63	122,3	121,01	120,95	126,86
Processing (%)	0,51	54,57	41,13	75,87	86,11	93,23	92,48
Current (energy)	686,11	718,81	844,7	934,56	977,82	987,33	975,25
Orange Pi Plus 2							
Simultaneous calls	0	101	97	95	87	95	91
Memory (MB)	81,97	132,74	120,92	119,7	120,95	124,95	122,67
Processing (%)	0,56	100	99,99	99,99	99,98	99,99	99,99
Current (energy)	805,33	1154,52	1150,26	1145,42	1155,57	1159	1157,58

C. IAX2 calls

This subsection addresses the data collected using the IAX2 protocol. They are shown in Table VI. It is possible to note that the G711.a CODEC stands out with low memory consumption, processing, power and high number of simultaneous calls on the Raspberry Pi 3 and Banana Pi M3 devices. The Orange Pi Plus 2 was highlighted by the

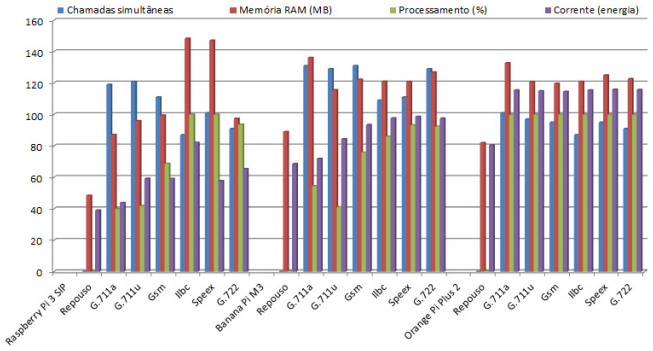


Fig. 5. Comparative SIP with Transcoding.

GSM CODEC due to the high number of simultaneous calls, low processing, power consumption and moderate memory consumption. However, if compared to the other devices addressed in this research, this of is the one that has inferior performance as shown in Fig.6.

TABLE VI
DATA COLLECTED WITH THE IAX2 PROTOCOL.

Raspberry Pi 3 IAX2							
Codec's	Repouso	Alaw	Ulaw	Gsm	Ilbc	Speex	G.722
Simultaneous calls	0	181	163	171	85	75	165
Memory (MB)	48,48	89,54	83,66	78,16	78,99	78,38	82,49
Processing (%)	0,05	52,28	50,67	86,41	95,21	86,64	72,23
Current (energy)	39,076	67,284	75,122	58,587	73,816	76,871	71,732
Banana Pi M3 IAX2							
Simultaneous calls	0	201	191	201	105	105	181
Memory (MB)	89,04	107,09	107,62	108,77	101,94	103,04	106,08
Processing (%)	0,51	59,82	61,47	47,97	68,06	78,69	53,46
Current (energy)	68,611	83,392	85,964	82,92	92,7	95,299	86,695
Orange Pi Plus 2 IAX2							
Simultaneous calls	0	201	189	201	81	45	91
Memory (MB)	81,97	107,23	105,25	101,09	97,29	91,97	102
Processing (%)	0,56	90,3	89,77	69,78	94	75,27	94
Current (energy)	80,533	110,878	110,31	106,29	116,047	95,658	93,499

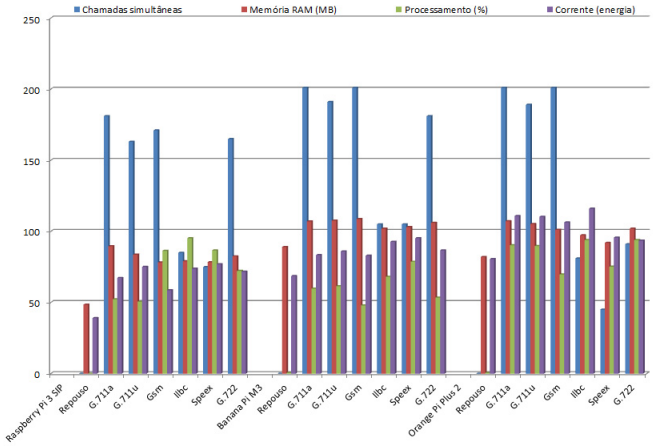


Fig. 6. Comparative analysis IAX2.

D. IAX2 calls with Transcoding

Table VII shows data collected using the IAX2 protocol with transcoding. Of immediately, we note there is a low number of concurrent calls supported. Given this context, it is possible to state that the embedded devices addressed

in this research do not support the process of transcoding CODEC's with the IAX2 protocol. Only in this test the Orange Pi Plus 2 device did not report high processing consumption, as shown in Fig.7.

TABLE VII
DATA COLLECTED WITH THE IAX2 PROTOCOL WITH TRANSCODING.

Raspberry Pi 3							
Codec's	**Repouso**	**Alaw**	**Ulaw**	**Gsm**	**Ilbc**	**Speex**	**G.722**
Simultaneous calls	0	13	14	13	14	14	16
Memory (MB)	48,48	70,81	70,43	70,43	83,48	69,95	84,02
Processing (%)	0,05	14,98	21,46	33,35	29,92	23,3	37,99
Current (energy)	390,76	389,51	392,62	403,92	467,55	498,12	463
Banana Pi M3							
Simultaneous calls	0	16	14	13	15	12	15
Memory (MB)	89,04	94,18	93,79	93,66	137,07	143,37	98,33
Processing (%)	0,51	27,34	26,44	32,18	28,42	39,57	31,74
Current (energy)	686,11	725,73	716,77	708,26	758,22	823,97	697,62
Orange Pi Plus 2							
Simultaneous calls	0	14	12	25	15	15	15
Memory (MB)	81,97	95,12	89,29	94,15	91,29	93,15	92,26
Processing (%)	0,56	25,32	20,06	32,92	31,63	22,05	47,52
Current (energy)	805,33	846,61	817,15	879,85	874,27	825,3	873,12

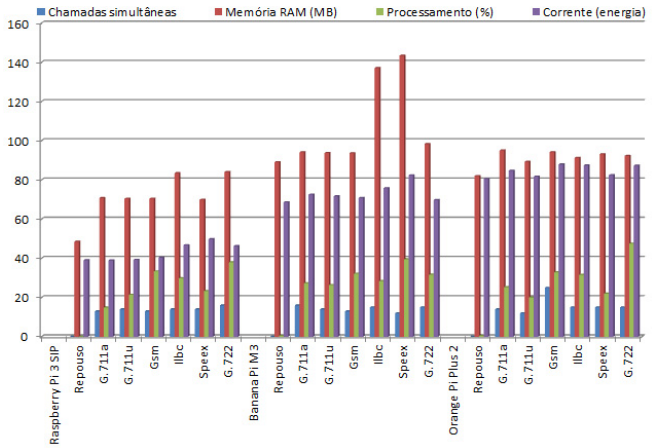


Fig. 7. Comparative analysis IAX2.

V. Conclusion and Future Work

In this work, an approach was performed to analyze performance and efficiency in three state-of-the-art embedded devices using the by voice over IP Asterisk, which measured the number of concurrent calls supported on each device with the SIP and IAX2 protocol, both in normal calls and in transcoded calls. In parallel, we performed the analysis of the consumption of RAM memory, processing and energy.

The prototyping was performed to compare the three embedded devices using the Asterisk. The results were surprising. The Raspberry Pi 3 and Banana Pi M3 devices satisfactorily support a high number of simultaneous calls with memory, processing and moderate power through CODEC G.711a and G.711u. However, the Orange Pi Plus 2 device showed high processing power. Thus, it is assert that this device is not suitable for use with Asterisk.

All 3 devices showed stability throughout the search. In no time occurring unintentional restart of the equipment, even during loads.

As future works, it is proposed to perform these experiments with the Codec Opus, since it was not possible to perform the compilation due to incompatibility of the embedded device. It is possible also perform the same approach with the IAX2 protocol using encryption.

References

[1] P. S. M. Bernal, *Voz sobre protocolo IP: A nova realidade da telefonia*, São Paulo, Brazil: Erica, 2007.

[2] S. Colcher, A. T. A. Gomes, A. O. Silva, G. L. S. Filho and L. F. G. Soares, *VoIP: voz sobre IP*, Rio de Janeiro, Brazil: Elsevier, 2005.

[3] I. I. Androulidakis, *Mobile Phone Security and Forensics: A Practical Approach*, Springer, 2016.

[4] A. Sulkin, *PBX systems for IP telephony*, New York, NY, United States, McGraw-Hill, 2002.

[5] R. Bryant, L. Madsen and J. V. Meggelen *Asterisk: The definitive guide*, 4 ed. United States: O'Reilly Media, 2013.

[6] D. Villacis, A. R. Freddy and L. A. R. Cueva *Performance analysis of VoIP Services over WiFi-based systems*, In: Communications and Computing (COLCOM) IEEE Colombian Conference on. IEEE, 2013. p. 1-6.

[7] N. M. Edan, S. Turner, A. Al-Sherbaz and S. Ajit *Performance evaluation of QoS using SIP and IAX2 VVoIP protocols with CODECS*, In: Communications and Computing (COLCOM) IEEE Colombian Conference on. IEEE, 2013. p. 1-6.

[8] A. D. Tesfamicael, V. Liu, W. Caelli and J. Zureo *Implementation and evaluation of open source unified communications for SMBs*, In: Computational Intelligence and Communication Networks (CICN), 2014 International Conference on. IEEE, 2014. p. 1243-1248.

[9] ABID, F. et al. *Embedded implementation of an IP-PBX /VoIP gateway. Proceedings of the International Conference on Microelectronics, ICM, n. Icm, p. 5?8, 2012. ISSN 2159-1660.*, (2012)

[10] R. Jain *The Art of Computer Systems Performance Analysis: techniques for experimental design, measurement, simulation, and modeling*, John Wiley and Sons, 1991.

[11] Digium *Dimensioning*, (2017) Available: http://www.digium.com/blog/2012/09/25/asterisk-dimensioning-what-server-do-i-need

[12] RFC5456 *Inter-Asterisk eXchange Version 2*, (2017) Available: https://tools.ietf.org/html/rfc5456

APÊNDICE D – A Ser Submetido

An Approach to Performance Analysis and Efficiency Power on Embedded Devices Using Asterisk

Adauto Cavalcante Menezes*, Toniclay Andrade Nogueira†,
Edward David Moreno Ordonez‡ and Admilson de Ribamar Lima Ribeiro§
Computing Department
UFS - Segipe Federal University, Aracaju, Sergipe
E-mail: *adauto.cavalcant@gmail.com, †toniclay@globo.com, ‡edwdavid@gmail.com, §admilson@ufs.br

***Abstract*—Voice over IP (VoIP) communication will dominate the computing world for years to come. In order to perform VoIP communication, it is necessary to encode and decode the voice. This process consumes the main computational resources, as an example, we have the processor and memory. The telecommunication industries provide equipment with high values, which makes access to this technology still very restricted. Embedded devices are purposely constructed devices for certain applications, they execute systems with high criticality complex. Asterisk is a free software for voice over IP communication and has as main function to implement the functions of a telephone exchange. These technologies promise to reduce costs and maximize results. This work performs a performance analysis on three modern embedded devices (Raspberry Pi 3, Orange Pi Plus 2 and Banana Pi M3) using the Asterisk voice over IP communication system. Performance analysis consists of evaluate the jitter, delay and bandwidth, as well as the number of concurrent calls supported in each device with SIP and IAX2 protocols with CODEC's G.711a, G.711u, Gsm, Speex, Ilbc, G.722, and in parallel, monitor the RAM consumption, processing and energy. The results show that the Raspberry Pi 3 and Banana Pi M3 devices support in a satisfactory manner a high number of simultaneous calls with memory consumption, processing and moderate energy. However, the Orange Pi Plus 2 device showed high consumption of the processing.**

***Index Terms*—VoIP, Asterisk, Embedded Devices, Performance Analysis, Embedded Systems.**

I. Introduction

The term Voice over Internet Protocol (VoIP) is conceptualized as the voice communication in networks that use the Internet Protocol (IP), which developed with the appearance of IP Telephony, consists of the provision of telephony services using the network IP for the establishment of calls and voice communication [1]. In the middle of the year 1990, the definition of VoIP was consolidated, when the Internet Phone from VocalTec Communications, the first commercial software that enabled the communication of voice over IP, but with poor communication quality [2].

For Androulidakis [3], private telephone exchanges serve to communicate between internal telephones and communication with the public telephone network. There are IP communication systems or also called Private Branch Exchange (PBX) IP and Time Division Multiplexing (TDM) communication systems or conventional TDM PBXs. Their software can be offered proprietary or through open source. Communication with the external medium depends on the interface for connecting analog or digital lines, usually provided by telecommunication operators, and communication with internal telephones depends exclusively on the central office itself. Initially, the main advantage of the use of PBX communication systems was the cost of calling the internal lines, as there is an internal switching of circuits, which makes the call free.

Also according to [3], voice communication systems have gained popularity and now have functionalities, services that were not available from telecom operators such as hunt groups, call forwarding, and dial-by-extension. According to [4], the trend is to migrate to IP telephony.

In the year of 1999 came the software for communication of voice over IP Asterisk. According to [5], Asterisk is free software that performs all the functions of a conventional telephone exchange, developed by Digium. Currently, it receives numerous contributions from developers around the world, as it is a promising area of application that is constantly in development.

The telecommunication industries provide equipment with high values and mostly proprietary equipment. This makes it difficult to access VoIP technology.

Thus, it is necessary to provide a mechanism to reduce expenses in the area of telephony, preferably with characteristics similar to that of a conventional telephone exchange, it is also necessary to expand and encourage the knowledge of students and researchers in the area of information technology and telecommunications. In this context, the importance of efficiency and cost reduction of the new generation of revolutionary personalized systems for voice over IP communication is justified. Thus, it is reinforced the idea that any advance of quality in this technology can propitiate a considerable increase in the quality of the voice communication, both in academic environment, as in the telecommunications industry, thus, we have the motivation that aims to accomplish this work.

From Asterisk, it is possible to implement a voice communication system on embedded, low-cost devices that provides the necessary mechanisms of a conventional tele-

phone exchange. For this, a performance analysis should be performed on three modern embedded devices (Raspberry Pi 3, Orange Pi Plus 2 and Banana Pi M3), using the Asterisk voice over IP communication system. For this, we must perform an analysis of the behavior of IP phone calls on the devices through research jitter, delay and bandwidth, as measuring the number of concurrent calls supported on each device with the SIP and IAX2 protocols with different CODEC's and in parallel monitoring the RAM, processing and power consumption. At the end we can see which implementation has performed best to support the Asterisk voice over IP communication system.

II. Related Work

In this section, all works classified as significant and related to the present study are discussed. A systematic analysis was carried out in order to find relevant works in bases considered important in the area of computing (IEEE Xplore, Science Direct and the Brazilian Digital Library of Computation). The amount of work that Asterisk addresses in embedded devices is greatly reduced. In this way, work on performance analysis with the use of the SIP and IAX2 protocol was also researched, thus, it was possible to know the measurement techniques currently used.

In a paper entitled "Performance Analysis of VoIP Services over WiFi-based systems", the authors [6] presented an Alix hardware performance analysis for usage in VoIP systems under Wireless Fidelity (Wifi) networks, adopts a server with Embedded Asterisk software. This analysis consists of tests of the central processing unit (CPU) and RAM memory with a fixed number of simultaneous calls with the SIP and IAX2 protocols, and with the CODEC'S GSM, G.711 u / a, SPEEX and G.726. Still in [6], the authors could perform an energy efficiency analysis at the time of high number of simultaneous calls, as well as conduct simultaneous call behavior tests in an interconnection between Asterisk servers, this way making a more robust research.

In [7] present in their article a performance evaluation and Quality of Service (QOS) multimedia transmission (voice and video), using the SIP and IAX2 protocols based on an Asterisk server. The quality of service evaluated used some parameters of Qos, such as bandwidth, jitter and delay, in order to investigate the performance of different CODECS of voice and video. In the work of [7], the authors could conduct concurrent call behavior tests with both softphones and servers. They could also design and develop an application for video transmission with support for the IAX2 protocol, in order to allow the completion of one of their tests listed in this work.

In the article "Implementation and Evaluation of Open Source Unified Communications for SMBs", the authors [8] presented the implementation and evaluation of a unified communication system composed of instant messaging and sharing (voice and video), voice messaging, VoIP communication and mobility. The evaluation covered only quantitative measurements of instantaneously supported telephone calls. However, the authors [8] started their work with a unified communication system approach, nevertheless, in their experiments they performed only voice and video communication tests, thus, they did not demonstrate the efficiency of the system when used with several modules of the proposed unified communication system. Also in [8], the authors carried out a work with great contribution as for example the accomplishment of these same ones for the industry, researchers and academic community, however, there is still much to be done, experiments in embedded hardware devices, as well as the measurement of energy efficiency at times of high system consumption, are not addressed in this research.

In [9], their article "Embedded Implementation of an IP-PBX / VoIP Gateway", propose the idea of designing and implementing an embedded PBX-IP gateway, which uses low cost and open source solutions. The system integrates FPGA hardware and incorporates software into external memory. Their experiment consists of an ML501 FPGA hardware, with embedded asterisk software and two pcx86 computers, one connected to serial port and the other connected to the ethernet network. The authors [9], have performed only a test of Asterisk software support in the hardware embedded FPGA. The authors performed only one network connectivity test. However, they did not perform tests to gauge the number of concurrent calls supported, measurement of power consumption, processing and memory.

The papers presented above are of relevant content, they address an investigative practice for the concept of performance evaluation of a VoIP communication system that uses the SIP and IAX protocols.

Table I shows the hardwares, softwares, protocols and CODECS used in the experiments of the related works, as well as in our work. Table II shows a comparison between the activities carried out in our work and those presented in the related works. It is observed that our is more complete. In the following sections, the implementation details of performance analysis and energy efficiency in embedded devices using Asterisk, as well as prototyping and the measurements taken to extract the results will be described.

It's worth noticing the use of various versions of the Linux operating system, as well as the use of Asterisk software in embedded device and in different hardware. In addition, all the works performed experiments in real environment, which makes the research more productive. The authors [8] and [6] addressed the use of the SIPP tool, which makes it possible to make several simultaneous calls. The SIP protocol, predominant in the three studies, was used more frequently, as well as CODEC G.711.

A good research work requires parameters for validating results. However, all the above mencioned authors did not address any metrics for validation, which leads to

TABLE I
Data of Related Works

Authors	Hardware	Software	Protocol	Codec
[6]	PC X86 i7 8gb RAM Switch Iphone	AsteriskNow X-lite Zoiper Wireshark	SIP IAX2	G.711 u G711a Gsm G.722 Speex H.263 H.264 H.261 H.263p
[7]	PC x86 i7 3.40Ghz 8GB RAM	CentOS Ubuntu FreePBX Sipp Openfire	SIP	G.711 u G.711a G.729 Gsm
[8]	Alix 2d2	Voyage Asterisk Sipp	SIP IAX2	G.711 u G.711a G.722 Gsm Speex G.726
[9]	ML501 FPGA	Linux Asterisk	-	-
This work	Raspberry Pi 3, Banana Pi M3, Orange Pi Plus 2, Mikrotik 951g.	Raspbian Armbian Wireshark Zabbix Zoiper X-lite	SIP IAX2	G.711a G.711u G.722 Gsm Ilbc Speex

TABLE II
Activities carried

Qualities	[6]	[7]	[8]	[9]	This work
Embedded device	x	-	-	x	x
Simultaneous calls sip	x	-	x	-	x
Simultaneous calls iax2	x	-	-	-	x
Memory Consumption	x	-	-	-	x
CPU consumption	x	-	x	-	x
Energy consumption	-	-	-	-	x
Validation metric	-	-	-	-	x

believe in the occurrence of possible errors in the results demonstrated.

III. Methodology and Approach

When defining a performance evaluation methodology, care must be taken for not to make common mistakes, such as lack of objectives, tendentious proposals, incorrect methods of evaluation, among others. To avoid such errors, the ideal is to adopt a systematic approach such as the one proposed by [10], which was applied in this work. To employ this methodology it becomes necessary to follow a sequence of steps.

The first step is the definition of objectives and the system. The second step is the preparation of the list of expected services and results. The third step is the selection of metrics, which establish the criteria for the comparison of performance. The fourth step is to compile the parameter list, in fact it is the list of parameters that affect performance. The fifth step, about the choice of factors for study, these factors are parameters that will suffer variations during the research. The sixth step is the selection of the evaluation technique, there are three techniques, which are simulation, analytical modeling and measurement. As a seventh step, you have the choice of the load, which consists of a list of service requests to the system. It is important to portray the actual use.

Then the eighth step is the planning of the experiments. As a ninth step, we have to analyze and interpret the data, in this step we must use adequate statistical techniques in order to consolidate the results obtained, in order to allow conclusions about the performance of the system. Already tenth and final step is the presentation of the results, in this step we must pay attention to the final presentation of the evaluation.

A. Application of the Methodology

In applying this methodology, we notice its importance, given the organized form that the work was conducted. Initially, it is necessary to define the objectives, then the scope of the system, the services offered, as well as the evaluation technique, which are shown below.

Goals:

- Perform a performance analysis on three low-cost embedded devices;
- Determine relevant factors in the performance of these equipments.

System:

- The system corresponds to a voice communication software over IP called Asterisk, it interacts with the medium through the reception and realization of telephone calls through IP Protocol.

Service:

- Voice over IP communication.

Assessment Technique:

- Measurement, therefore, is a useful technique for analyzing the performance of computer systems.

The activity was divided into five stages:

- Step 1 - Design and test scenario;
- Step 2 - Specification of metrics;
- Step 3 - Definition of parameters, factors and load;
- Step 4 - Planning and conducting the experiments;
- Step 5 - Statistical analysis of the results obtained.

The next subsection describes the first three steps, while the fourth and fifth are discussed in the Experiments and Results section.

B. Design and Test Scenario

To carry out the prototyping, we must assume the existence of a computational model identical to the real production environment. In the literature, good descriptions for performance analysis in voice over IP communication systems have been found, for example, the work of [6] and [7]. The first work presents a performance analysis

of the Alix 2D2 hardware for use in VoIP systems. The second one presents a performance assessment and Quality of Service (QOS) for multimedia transmission (voice and video).

Based on these works and technical specifications of commercially available equipment, the present work was carried out with three modern embedded devices and with the proposal of complementing the research already done, in this way, it effectively contributes to the telecommunications industry, for small and large companies, as well as, for the academic area, therefore, extends the realization of new researches.

This work verifies the possibility of verifying the use of modern embedded devices as servers of a voice over IP communication system called Asterisk using the SIP and IAX2 protocols and with the following CODEC'S: G.711a (Alaw), G.711u (Ulaw), Gsm, Speex, Ilbc and G.722, no other CODECs were used because of the limitations found in the work performed, such as free softphones used only to support these CODEC'S. The embedded devices discussed in this work have an average price of 50 dollars, thus, it is possible to consider as a solution of low cost.

The performance analysis consists in verifying of evaluate the jitter, delay and bandwidth, as well as the number of concurrent calls supported in each embedded device with the protocols and CODEC's mentioned above and in parallel, carry out the monitoring of the consumption of RAM, processing and energy. For this, the tool was used for traffic analysis wireshark, software specialized in traffic analysis in IP networks, widely used by researchers in the academic environment, for example, [7].

In order to perform this work, it was necessary to elaborate a test scenario, so the scenario included three modern and low cost embedded devices (Raspberry Pi 3, Banana Pi M3 and Orange Pi Plus 2), a RouterBoard 951g Mikrotik that made the switch , a Core i7 8GB RAM 500GB HD laptop. Fig.1 illustrates the architecture of the scenario designed to perform the experiments.

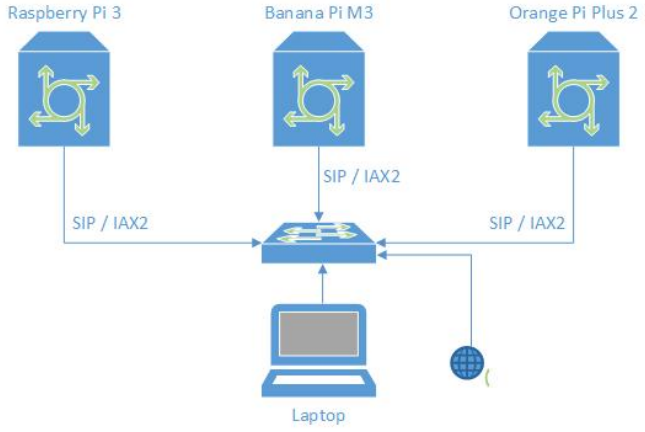


Fig. 1. Test scenario.

The RouterBoard Mikrotik that realizes the switch is responsible for interconnecting all the equipment in network. Each embedded device supports the Asterisk voice over IP communication system. The laptop has been allocated to perform the data collection with the Wireshark software.

Currently there is a great offer of embedded devices in the market, also called SBC (Single-Board Computers), with the most diverse configurations, for example, it is possible to mention Raspberry Pi Zero, Raspberry 2 Model B, Raspberry Pi 3 Model B, among others. Table III illustrates four models of last generation embedded devices, their configurations and price.

TABLE III
EMBEDDED DEVICES

Manufacturer	**Raspberry Pi**	**Banana Pi**	**Orange Pi**	**Odroid**
Model	3 Model B	M3	Plus 2	XU4
CPU Cores	4	8	4	4+4
CPU Freq.	1.2GHz	2GHz	1.5GHz	2.1GHz 1.5GHz
Memory	1GB DDR2	2GB DDR3	2GB DDR3	2GB DDR3
Storage	MicroSD	MicroSD USB Sata 2.0	MicroSD USB Sata 2.0	MicroSD eMMC
Linux	Yes	Yes	Yes	Yes
Price	$40	$59,99	$59,99	$59

According to [11], it is not easy to size a minimum hardware for the installation of the voice over IP communication system Asterisk, however, for this difficult question there is no precise answer. It is recommended to always use a hardware slightly beyond the needs, also suggests that if the system needs to support more than 20 simultaneous calls, a dedicated server should be used. Thus, for the accomplishment of this work, were chosen three devices of low cost and with good configurations, they are: Raspberry Pi 3, Banana Pi M3 and Orange Pi Plus 2.

IV. Experiments and Results

This section presents the implementation of the experiment, which consists of: assembling the test scenario with the embedded devices individually; accomplish software approach in the devices for the elaboration of the experiment; proceed with the process of collecting the jitter, delay and bandwidth with the Wireshark software; implement a code in Shell Script in order to generate loads of SIP and IAX calls with CODEC's G.711u (Ulaw), G.711a (Alaw); Gsm, Speex, Ilbc and G.722; run high call flows and in parallel carry out the monitoring of the consumption of processing; RAM and energy; and finally, to gauge the results. Fig.2 illustrates the actual testing scenario with the Banana Pi M3 device.

The tests were performed in order to reach the maximum number of calls supported with no occurrence of errors. Thus, several attempts were made to until you

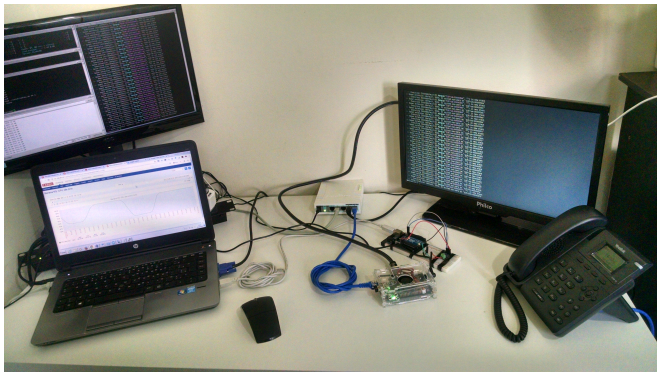


Fig. 2. Experiment running.

find the expected result. Thereafter, 10 replicates were performed to validate the occurrence of no error. For it is a static result, there is no variation of the mean and thus a greater number of repetitions can be dispensed with.

The collect was performed with the protocol SIP and IAX2, and with the Codec's supported by the free softphones addressed (Zoiper and X-lite). However, 33 collections (6 SIP and 5 IAX2 in each embedded device) were performed, each of calls lasting approximately 2 minutes, so it was possible to obtain a good quantitative of SIP and IAX2 packets to carry out the analysis, approximately 5,000 datagrams IP.

The Wireshark tool has filters that automatically perform jitter, bandwidth and delay analysis. Though, the mean was used as a parameter for the measurements. Notwithstanding, the mean bandwidth and delay were calculated manually by exporting the captured data to LibreOffice Calc, since Wireshark does not report the average, only the maximum value reached.

Thus, calls were made between the softphones and a self-service set up in the Asterisk of the embedded device. The tests were performed with the SIP and IAX2 protocols, as well as with the CODEC's mentioned above, in order to achieve the objective of this work with the measurements.

In order to carry out the generation of simultaneous SIP call load, Asterisk was used in a virtual machine installed on the laptop, and a dialer was implemented in Shell Script. For calls with the IAX2 protocol, it was held the exchange of the channel on the dialer, from SIP to IAX2. The dialer code is shown in Fig.3.

Thus, two files were generated to perform high numbers called simultaneous, called CallFilePlay.sh and CallFileCodec.sh. The first for normal calls, the second for calls with transcoding. The only difference between the two files is that in CallFileCodec.sh the extension number on the dialer is different. In this way, these files were inserted in the Asterisk / etc / Asterisk directory on the dialer virtual machine. And so, it was possible to start the tests.

It was also necessary to configure an audio file called test.gsm, with 3 minutes, and include it in Asterisk's

```
#!/bin/bash
aux1=$1;
aux2=$2;
CONTADOR=0
CONTADOR2=0
echo "Enviando $1 chamadas $2 vezes"

while [ $CONTADOR2 -lt $aux2 ]; do
echo "$CONTADOR2 x"
while [ $CONTADOR -lt $aux1 ]; do
echo "Channel: SIP/embarcado/5000" >>
/etc/asterisk/callfiles/teste$CONTADOR_$CONTADOR2.call
echo "CallerID: embarcado" >>
 /etc/asterisk/callfiles/teste$CONTADOR_$CONTADOR2.call
echo "MaxRetries: 0" >>
/etc/asterisk/callfiles/teste$CONTADOR_$CONTADOR2.call
echo "RetryTime: 60" >>
 /etc/asterisk/callfiles/teste$CONTADOR_$CONTADOR2.call
echo "WaitTime: 30" >>
/etc/asterisk/callfiles/teste$CONTADOR_$CONTADOR2.call
echo "Context: default" >>
/etc/asterisk/callfiles/teste$CONTADOR_$CONTADOR2.call
echo "Extension: 3000" >>
/etc/asterisk/callfiles/teste$CONTADOR_$CONTADOR2.call

mv /etc/asterisk/callfiles/teste$CONTADOR_$CONTADOR2.call
/var/spool/asterisk/outgoing/
let CONTADOR=CONTADOR+1;
done
sleep 10s;
CONTADOR=0
let CONTADOR2=CONTADOR2+1;
done
```

Fig. 3. Dialer code.

"sounds / var / lib / asterisk / sounds" directory, whose purpose is for Asterisk to receive calls from the load generator, forward them to self-service and then play the audio. In this way, occurs the media flow limited time, enough to run the concurrent call test.

To start the tests, the virtual machine was used on the laptop to run the dialer load generator, and in parallel the monitoring of the consumption of RAM memory and processing with Zabbix software; monitoring the energy consumption with the circuit INA219, as well as verification of errors occurring in the processing of calls, through of the debug in Asterisk.

A Yalink T19p E2 IP telephone was also used in order to validate the quality of the call at high load consumption. However, it was not possible to validate the quality with the IAX2 protocol and neither with the Speex and Gsm CODEC's with the SIP protocol, due to the license limitations of the telephone set.

As the tests were carried out, the CODECs were modified, as well as the protocol. The experiments of the simultaneos calls were divided into 4 stages, which will be described of in next subsections, as well as their results.

The following subsections demonstrate the results of the jitter, delay and bandwidth collected with the SIP and IAX2 protocols on the embedded devices Raspberry Pi 3, Banana Pi M3 and Orange Pi Plus 2, with mean, standard deviation and confidence interval of 95%, as well as, holds brief discussions.

TABLE IV
BANDWIDTH SIP

Raspberry Pi 3				
Codec's	**Average**	**Standard Deviation**	**Inferior Limit**	**Upper Limit**
G.711a	81,13	0,76	81,08	81,18
G.711u	81,11	0,77	81,06	81,15
Gsm	29,57	0,28	29,56	29,58
Speex	24,52	0,20	24,51	24,52
Ilbc	24,36	0,38	24,35	24,38
G.722	81,09	0,75	81,06	81,11
Banana Pi M3				
Codec's	**Average**	**Standard Deviation**	**Inferior Limit**	**Upper Limit**
G.711a	80,50	0,73	80,48	80,53
G.711u	80,48	0,72	80,46	80,5
Gsm	29,38	0,27	29,37	29,39
Speex	24,58	0,24	24,58	24,59
Ilbc	24,49	0,00	24,49	24,49
G.722	80,94	0,79	80,92	80,97
Orange Pi Plus 2				
Codec's	**Average**	**Standard Deviation**	**Inferior Limit**	**Upper Limit**
G.711a	80,51	0,74	80,49	80,53
G.711u	80,52	0,74	80,50	80,54
Gsm	29,40	0,27	29,39	29,41
Speex	24,58	0,23	24,57	24,58
Ilbc	24,49	0,00	24,49	24,49
G.722	81,06	0,76	81,04	81,08

TABLE V
BANDWIDTH IAX2

Raspberry Pi 3				
Codec's	**Average**	**Standard Deviation**	**Inferior Limit**	**Upper Limit**
G.711a	74,21	0,72	74,19	74,23
G.711u	74,20	0,72	74,18	74,22
Gsm	22,99	0,22	22,98	23,00
Speex	18,14	0,17	18,13	18,14
Ilbc	20,13	0,02	20,13	20,13
Banana Pi M3				
Codec's	**Average**	**Standard Deviation**	**Inferior Limit**	**Upper Limit**
G.711a	74,37	0,78	74,35	74,4
G.711u	74,33	0,78	74,31	74,36
Gsm	23,03	0,23	23,03	23,04
Speex	18,18	0,18	18,17	18,18
Ilbc	20,03	0,32	20,02	20,04
Orange Pi Plus 2				
Codec's	**Average**	**Standard Deviation**	**Inferior Limit**	**Upper Limit**
G.711a	74,57	0,70	74,55	74,60
G.711u	74,25	0,73	74,23	74,28
Gsm	23,01	0,23	23,00	23,02
Speex	18,16	0,18	18,15	18,16
Ilbc	20,13	0,19	20,12	20,13

A. Bandwidth

Tables III and IV illustrate the bandwidth of the collected packets. We note that CODEC's G.711a, G.711u and G.722 have higher bandwidth rates and identical values, since CODEC's Gsm, Speex and Ilbc stand out because they have lower bandwidth, both with SIP protocol and with IAX2. The results found on the three embedded devices are similar. However, the distinction is clear in the approach of the SIP and IAX2 protocols, since it is observed that the IAX2 protocol consumes a smaller band in front of the SIP, as shown in Fig.4. It was not possible to collect the CODEC G.722 with the protocol IAX2 for not finding free softphone with CODEC support. Moreover, it is possible to state that the standard deviations and the confidence interval are minimal due to the high number of samples to carry out the approach.

In order to facilitate the identification of the devices in Fig.4, a code predecessor to the CODEC'S name has been inserted, so that, (R.) for the Raspberry Pi device 3, (B.) for the Banana Pi device M3 and (O.) , respectively for the Orange Pi Plus 2 device.

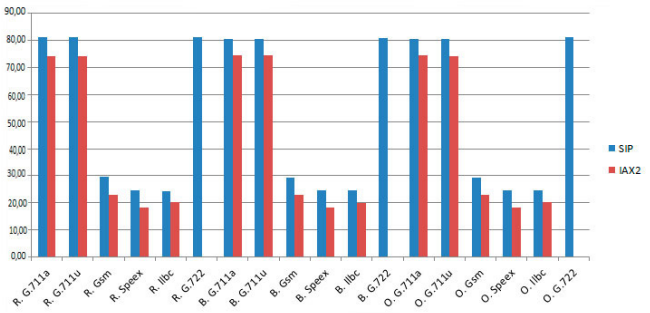


Fig. 4. Bandwidth SIP e IAX2.

B. Delay

TABLE VI
DELAY SIP

Raspberry Pi 3				
Codec's	**Average**	**Standard Deviation**	**Inferior Limit**	**Upper Limit**
G.711a	20,02	0,30	20,00	20,04
G.711u	20,02	0,30	20,00	20,04
Gsm	20,01	0,12	20,01	20,02
Speex	20,02	0,12	20,01	20,03
Ilbc	29,92	0,25	29,91	29,93
G.722	20,00	0,30	19,99	20,01
Banana Pi M3				
Codec's	**Average**	**Standard Deviation**	**Inferior Limit**	**Upper Limit**
G.711a	20,00	0,06	20,00	20,01
G.711u	20,00	0,05	20,00	20,01
Gsm	20,00	0,06	20,00	20,01
Speex	20,00	0,12	20,00	20,01
Ilbc	30,01	0,41	29,99	30,02
G.722	20,00	0,27	20,00	20,01
Orange Pi Plus 2				
Codec's	**Average**	**Standard Deviation**	**Inferior Limit**	**Upper Limit**
G.711a	20,00	0,06	20,00	20,01
G.711u	20,00	0,07	20,00	20,01
Gsm	20,00	0,11	20,00	20,01
Speex	20,00	0,12	20,00	20,01
Ilbc	30,01	0,17	30,00	30,01
G.722	20,00	0,10	20,00	20,01

Tables V and VI demonstrate the delay of the collected packets. It is possible to observe that Delay with the SIP

TABLE VII
DELAY IAX2

Raspberry Pi 3				
Codec's	**Average**	**Standard Deviation**	**Low Limit**	**Upper Limit**
G.711a	0,02	0,00	0,02	0,02
G.711u	0,02	0,00	0,02	0,02
Gsm	0,01	0,02	0,02	0,02
Speex	0,02	0,00	0,02	0,02
Ilbc	0,03	0,00	0,03	0,03
Banana Pi M3				
Codec's	**Average**	**Standard Deviation**	**Low Limit**	**Upper Limit**
G.711a	0,02	0,01	0,02	0,02
G.711u	0,02	0,01	0,02	0,02
Gsm	0,02	0,01	0,02	0,02
Speex	0,02	0,01	0,02	0,02
Ilbc	0,03	0,01	0,03	0,03
Orange Pi Plus 2				
Codec's	**Average**	**Standard Deviation**	**Low Limit**	**Upper Limit**
G.711a	0,02	0,00	0,02	0,02
G.711u	0,02	0,00	0,02	0,02
Gsm	0,02	0,00	0,02	0,02
Speex	0,02	0,00	0,02	0,02
Ilbc	0,03	0,00	0,03	0,03

protocol is in accordance with the standard [13], which establishes that the default delay of the RTP packet should be 20ms. However, there is an exception to the CODEC Ilbc, which stands out with a value of 50% higher than the others, as shown in Fig.5. This is due to its coding algorithm, which performs high compression of the audio and consequently increases the transmission delay.

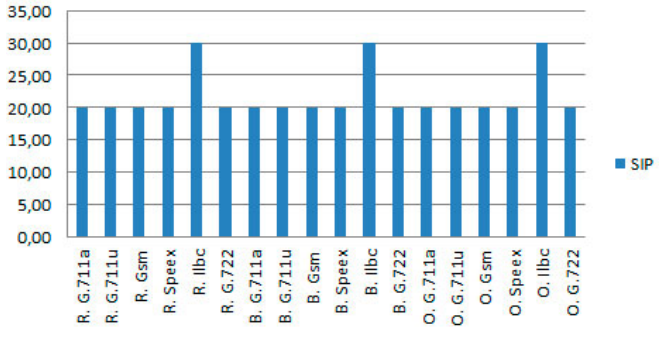


Fig. 5. Delay SIP.

Delay with the IAX2 protocol tends to zero with all CODEC's addressed in this research, which validates its proposal as a new protocol standard. According to [12] it is an open protocol that carries the transport of signaling and the media. In addition, the IAX2 protocol also proposes to eliminate any transmission delays. It is also observed that the standard deviations and the confidence interval are minimal due to the high number of samples to carry out the approach.

TABLE VIII
JITTER SIP

Raspberry Pi 3				
Codec's	**Average**	**Standard Deviation**	**Low Limit**	**Upper Limit**
G.711a	0,15	0,09	0,15	0,15
G.711u	0,15	0,09	0,15	0,15
Gsm	0,10	0,23	0,09	0,10
Speex	0,21	0,15	0,21	0,21
Ilbc	0,24	0,15	0,23	0,25
G.722	0,15	0,10	0,14	0,15
Banana Pi M3				
Codec's	**Average**	**Standard Deviation**	**Low Limit**	**Upper Limit**
G.711a	0,03	0,01	0,03	0,03
G.711u	0,03	0,01	0,03	0,03
Gsm	0,03	0,01	0,03	0,03
Speex	0,06	0,02	0,06	0,06
Ilbc	0,24	0,08	0,24	0,25
G.722	0,15	0,09	0,14	0,15
Orange Pi Plus 2				
Codec's	**Average**	**Standard Deviation**	**Low Limit**	**Upper Limit**
G.711a	0,04	0,01	0,04	0,04
G.711u	0,04	0,01	0,04	0,04
Gsm	0,06	0,03	0,06	0,06
Speex	0,06	0,02	0,06	0,06
Ilbc	0,09	0,04	0,10	0,10
G.722	0,06	0,03	0,06	0,06

TABLE IX
JITTER IAX2

Raspberry Pi 3				
Codec's	**Average**	**Standard Deviation**	**Low Limit**	**Upper Limit**
G.711a	0,02	0,19	0,01	0,02
G.711u	0,02	0,19	0,01	0,02
Gsm	0,00	0,00	0,00	0,00
Speex	0,02	0,19	0,01	0,02
Ilbc	0,02	0,19	0,01	0,03
Banana Pi M3				
Codec's	**Average**	**Standard Deviation**	**Low Limit**	**Upper Limit**
G.711a	0,02	0,19	0,01	0,03
G.711u	0,02	0,19	0,01	0,03
Gsm	0,02	0,19	0,01	0,02
Speex	0,02	0,19	0,01	0,02
Ilbc	0,03	0,19	0,01	0,02
Orange Pi Plus 2				
Codec's	**Average**	**Standard Deviation**	**Low Limit**	**Upper Limit**
G.711a	0,02	0,19	0,01	0,02
G.711u	0,02	0,19	0,01	0,02
Gsm	0,02	0,19	0,01	0,02
Speex	0,02	0,19	0,01	0,02
Ilbc	0,02	0,19	0,01	0,02

C. Jitter

Tables VII and VIII illustrate the jitter of the collected packets. It is possible to observe that, with both the SIP protocol and the IAX2 protocol, jitter tends to zero, this is because both protocols have as strategy to keep the frames in a buffer, in order to allow the slower frames arrive in time to be played in the correct sequence. The higher the amount of jitter, the greater the number of frames in the

buffer in order to minimize jitter in VoIP calls.

D. SIP calls

Table IV illustrates the data collected using the SIP protocol. It is possible to observe that the GSM CODEC supported the largest number of calls in the Raspberry Pi 3 and Banana Pi M3 devices. However, there is a high consumption of RAM. Nevertheless, given the RAM capacity of the Raspberry Pi 3 device of 1GB RAM and Banana Pi M3 of 2GB, this result becomes insignificant. Though, the GSM CODEC has the lowest processing. The CODEC G.711a stands out because it is the CODEC that consumes less energy and supports a significant number of calls because it is an embedded device.

TABLE X
DATA COLLECTED USING THE SIP PROTOCOL.

Raspberry Pi 3 SIP							
CODEC's	**Repose**	**Alaw**	**Ulaw**	**Gsm**	**Ilbc**	**Speex**	**G.722**
Simultaneous calls	0	257	251	291	211	211	241
Memory (MB)	48,48	95,01	93,92	121,13	102	94,66	125,75
Processing (%)	0,05	64,36	70,55	55,72	100	100	78,89
Current (energy)	390,76	486,91	691,31	644,37	838,33	848,5	726,43
Banana Pi M3 SIP							
Simultaneous calls	0	261	251	305	229	221	249
Memory (MB)	89,04	149,62	139,13	180,09	142,13	142,68	145,57
Processing (%)	0,51	69,05	72,43	63,89	100	100	73,16
Current (energy)	686,11	890,39	950,45	934,21	1038,25	988,64	905
Orange Pi Plus 2 SIP							
Simultaneous calls	0	247	261	251	241	241	251
Memory (MB)	81,97	113,77	125,5	113,51	122,32	124,46	117,83
Processing (%)	0,56	98,97	99,82	99,48	98,67	98,53	99,34
Current (energy)	805,33	1133,69	1138,51	1119,26	1130,21	1135,64	1137,43

On the other hand the Orange Pi Plus 2 device, got high processing in all tests. Given this information, it is possible to state that this device is not ideal for use with Asterisk.

Fig.6 illustrates the behavior of the calls made on the 3 embedded devices, which allows better evaluation and comparison of the data. However, we note that the Orange Pi Plus 2 device has identical results with all CODEC's addressed in this research, which reinforces the thesis that this device is not adequate to use with the software Asterisk.

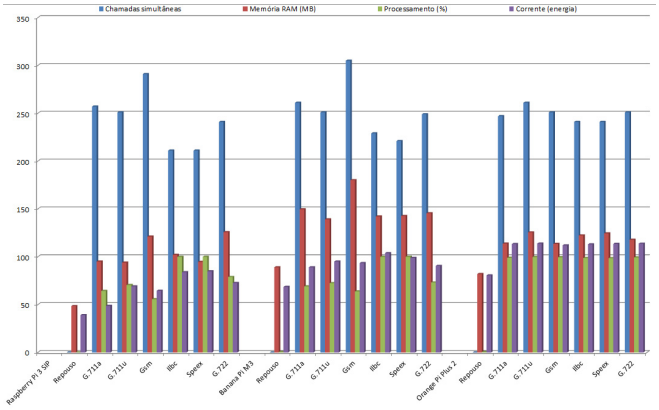


Fig. 6. SIP comparative analysis.

E. SIP calls with Transcoding

Table V shows the results collected on calls made with the SIP protocol and with transcoding. We note that with transcoding, the performance of the devices is greatly reduced. Even so, processing of the Orange Pi Plus 2 device remains high. This, once again, reinforces the claim that such a device is not suitable for use Asterisk. In the other devices, it is observed that the G.711a and G.711u CODECs have a higher number of concurrent calls supported and lower processing, memory and power consumption. In addition, the Raspberry Pi 3 device stands out with a considerable consumption of processing with CODEC's G.711a and G.711u, as shown in Fig.7.

TABLE XI
SIP WITH TRANSCODING.

Raspberry Pi 3							
CODEC's	**Repouso**	**Alaw**	**Ulaw**	**Gsm**	**Ilbc**	**Speex**	**G.722**
Simultaneous calls	0	119	121	111	87	101	91
Memory (MB)	48,48	87,17	95,98	99,47	148,3	247,16	97,49
Processing (%)	0,05	3,97	41,87	68,83	100	100	93,7
Current (energy)	390,76	438,85	592,64	592,04	821,49	577,56	654,27
Banana Pi M3							
Simultaneous calls	0	131	129	131	109	111	129
Memory (MB)	89,04	136,14	115,63	122,3	121,01	120,95	126,86
Processing (%)	0,51	54,57	41,13	75,87	86,11	93,23	92,48
Current (energy)	686,11	718,81	844,7	934,56	977,82	987,33	975,25
Orange Pi Plus 2							
Simultaneous calls	0	101	97	95	87	95	91
Memory (MB)	81,97	132,74	120,92	119,7	120,95	124,95	122,67
Processing (%)	0,56	100	99,99	99,99	99,98	99,99	99,99
Current (energy)	805,33	1154,52	1150,26	1145,42	1155,57	1159	1157,58

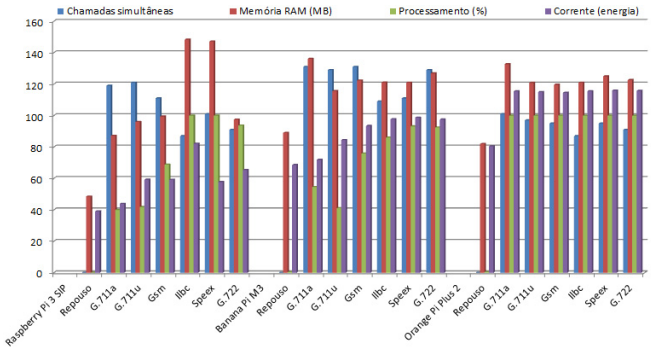


Fig. 7. Comparative SIP with Transcoding.

F. IAX2 calls

This subsection addresses the data collected using the IAX2 protocol. They are shown in Table VI. It is possible to note that the G711.a CODEC stands out with low memory consumption, processing, power and high number of simultaneous calls on the Raspberry Pi 3 and Banana Pi M3 devices. The Orange Pi Plus 2 was highlighted by the GSM CODEC due to the high number of simultaneous calls, low processing, power consumption and moderate memory consumption. However, if compared to the other devices addressed in this research, this of is the one that has inferior performance as shown in Fig.8.

TABLE XII
DATA COLLECTED WITH THE IAX2 PROTOCOL.

Raspberry Pi 3 IAX2							
Codec's	Repouso	Alaw	Ulaw	Gsm	Ilbc	Speex	G.722
Simultaneous calls	0	181	163	171	85	75	165
Memory (MB)	48,48	89,54	83,66	78,16	78,99	78,38	82,49
Processing (%)	0,05	52,28	50,67	86,41	95,21	86,64	72,23
Current (energy)	39,076	67,284	75,122	58,587	73,816	76,871	71,732
Banana Pi M3 IAX2							
Simultaneous calls	0	201	191	201	105	105	181
Memory (MB)	89,04	107,09	107,62	108,77	101,94	103,04	106,08
Processing (%)	0,51	59,82	61,47	47,97	68,06	78,69	53,46
Current (energy)	68,611	83,392	85,964	82,92	92,7	95,299	86,695
Orange Pi Plus 2 IAX2							
Simultaneous calls	0	201	189	201	81	45	91
Memory (MB)	81,97	107,23	105,25	101,09	97,29	91,97	102
Processing (%)	0,56	90,3	89,77	69,78	94	75,27	94
Current (energy)	80,533	110,878	110,31	106,29	116,047	95,658	93,499

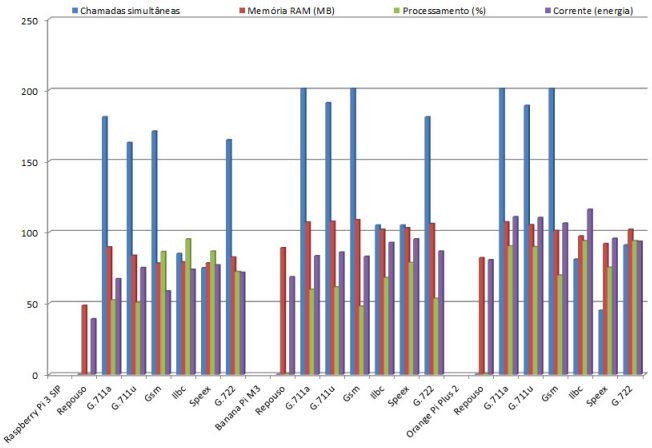


Fig. 8. Comparative analysis IAX2.

G. IAX2 calls with Transcoding

Table VII shows data collected using the IAX2 protocol with transcoding. Of immediately, we note there is a low number of concurrent calls supported. Given this context, it is possible to state that the embedded devices addressed in this research do not support the process of transcoding CODEC's with the IAX2 protocol. Only in this test the Orange Pi Plus 2 device did not report high processing consumption, as shown in Fig.9.

TABLE XIII
DATA COLLECTED WITH THE IAX2 PROTOCOL WITH TRANSCODING.

Raspberry Pi 3							
Codec's	**Repouso**	**Alaw**	**Ulaw**	**Gsm**	**Ilbc**	**Speex**	**G.722**
Simultaneous calls	0	13	14	13	14	14	16
Memory (MB)	48,48	70,81	70,43	70,43	83,48	69,95	84,02
Processing (%)	0,05	14,98	21,46	33,35	29,92	23,3	37,99
Current (energy)	390,76	389,51	392,62	403,92	467,55	498,12	463
Banana Pi M3							
Simultaneous calls	0	16	14	13	15	12	15
Memory (MB)	89,04	94,18	93,79	93,66	137,07	143,37	98,33
Processing (%)	0,51	27,34	26,44	32,18	28,42	39,57	31,74
Current (energy)	686,11	725,73	716,77	708,26	758,22	823,97	697,62
Orange Pi Plus 2							
Simultaneous calls	0	14	12	25	15	15	15
Memory (MB)	81,97	95,12	89,29	94,15	91,29	93,15	92,26
Processing (%)	0,56	25,32	20,06	32,92	31,63	22,05	47,52
Current (energy)	805,33	846,61	817,15	879,85	874,27	825,3	873,12

V. CONCLUSION AND FUTURE WORK

In this work, an approach was performed to analyze performance and efficiency energy in three state-of-the-art embedded devices using the software of voice over IP Asterisk, which measured the jitter, delay and bandwidth with SIP and IAX2 protocols with CODEC's G.711a (Alaw), G.711u (Ulaw), G.722, Ilbc, Speex and Gsm.

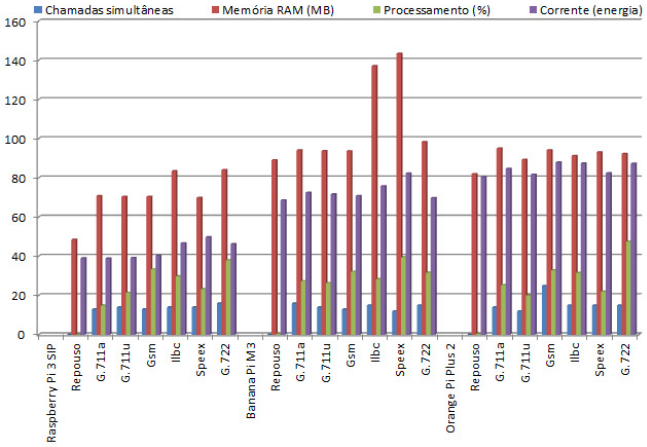


Fig. 9. Analysis IAX2 with Transconding.

The measurements were performed in order to compare the three embedded devices with use of Asterisk. However, the results showed great similarity in the data, both with the SIP protocol and with the IAX2 protocol, this in the network requirement.

We also verified the number of concurrent calls supported in each device with the SIP and IAX2 protocol, both in normal calls and in transcoded calls and in parallel, the analysis of the RAM memory consumption, processing and energy.

The prototyping was performed to compare the three embedded devices using the Asterisk. The results were surprising. The Raspberry Pi 3 and Banana Pi M3 devices satisfactorily support a high number of simultaneous calls with memory, processing and moderate power through CODEC G.711a and G.711u. However, the Orange Pi Plus 2 device showed high processing power. Thus, it is possible assert that this device is not suitable for use with Asterisk.

All 3 devices showed stability throughout the search. Not occurring unintentional restart of the equipment, even during high's loads.

The performance of the 3 embedded devices discussed in this work were evaluated with the purpose of finding the best device to support the communication system by voice over IP Asterisk. Nevertheless, new embedded devices, as well as new technologies will emerge. In this way, the possibility of extending this work is extended.

As future work, it is proposed to carry out these experiments with CODEC Opus, since it was not possible to perform the compilation on the embedded devices due to the incompatibility. It is also possible to carry out the same approach with protocol IAX2, using encryption. we can to propose a comparison of the behavior of the IAX2 protocol with transcoding on computing platforms with x86 archi-

tecture. Well how to perform the same bandwidth, jitter, and delay tests with other tools available in the market, specific to VoIP packet analysis, such as NetQuality Voip and SIP Tester.

References

[1] P. S. M. Bernal, *Voz sobre protocolo IP: A nova realidade da telefonia*, São Paulo, Brazil: Erica, 2007.

[2] S. Colcher, A. T. A. Gomes, A. O. Silva, G. L. S. Filho and L. F. G. Soares, *VoIP: voz sobre IP*, Rio de Janeiro, Brazil: Elsevier, 2005.

[3] I. I. Androulidakis, *Mobile Phone Security and Forensics: A Practical Approach*, Springer, 2016.

[4] A. Sulkin, *PBX systems for IP telephony*, New York, NY, United States, McGraw-Hill, 2002.

[5] R. Bryant, L. Madsen and J. V. Meggelen *Asterisk: The definitive guide*, 4 ed. United States: O'Reilly Media, 2013.

[6] D. Villacis, A. R. Freddy and L. A. R. Cueva *Performance analysis of VoIP Services over WiFi-based systems*, In: Communications and Computing (COLCOM) IEEE Colombian Conference on. IEEE, 2013. p. 1-6.

[7] N. M. Edan, S. Turner, A. Al-Sherbaz and S. Ajit *Performance evaluation of QoS using SIP and IAX2 VVoIP protocols with CODECS*, In: Communications and Computing (COLCOM) IEEE Colombian Conference on. IEEE, 2013. p. 1-6.

[8] A. D. Tesfamicael, V. Liu, W. Caelli and J. Zureo *Implementation and evaluation of open source unified communications for SMBs*, In: Computational Intelligence and Communication Networks (CICN), 2014 International Conference on. IEEE, 2014. p. 1243-1248.

[9] ABID, F. et al. *Embedded implementation of an IP-PBX /VoIP gateway. Proceedings of the International Conference on Microelectronics, ICM, n. Icm, p. 5?8, 2012. ISSN 2159-1660.*, (2012)

[10] R. Jain *The Art of Computer Systems Performance Analysis: techniques for experimental design, measurement, simulation, and modeling*, John Wiley and Sons, 1991.

[11] Digium *Dimensioning*, (2017) Available: http://www.digium.com/blog/2012/09/25/asterisk-dimensioning-what-server-do-i-need

[12] RFC5456 *Inter-Asterisk eXchange Version 2*, (2017) Available: https://tools.ietf.org/html/rfc5456

[13] RFC1890 *RTP Profile for Audio and Video Conferences with Minimal Control*, (2017) Available: https://tools.ietf.org/html/rfc1890