# PARECER SOBRE A VIABILIDADE TÉCNICO-JURÍDICA DAS ASSINATURAS DIGITAIS EM RELATÓRIOS DE EXAMES

EMENTA: Direito Civil. Certificação Digital. Infraestrutura de Chaves Públicas Brasileira. Código Civil Brasileiro. Medida Provisória 2.200-2/2001. RDC/ANVISA 302 de 13 de outubro de 2005. Lei 14.063/2020.

RELATÓRIO: Trata-se o expediente de uma consulta acerca da viabilidade técnico-jurídica da inserção de certificados digitais no padrão da Infraestrutura de Chaves Públicas Brasileira – ICP-Brasil – na liberação dos laudos laboratoriais, conforme proposto na Consulta Pública 912 da ANVISA – Agência Nacional de Vigilância Sanitária.

De acordo com o inciso II, do art. 3º, Seção III do Capítulo I – Das Disposições Iniciais a assinatura legalmente válida será apenas aquela realizada pelo responsável pela liberação do laudo laboratorial ou aquelas digitais inseridas por meio de certificado digital no padrão ICP-Brasil.

Ainda que consideremos a iniciativa bem-intencionada, apresentamos a seguir diversas razões que justificam a impossibilidade da adoção de tal padrão sob a perspectiva técnico-operacional, bem como a análise de diversos conceitos que devem compor a as variáveis norteadoras dos protocolos laboratoriais.

Esmiuçada a matéria, passamos à explanação.

## O QUE É LAUDO LABORATORIAL E O QUE É LAB REPORT? QUAIS AS DIFERENÇAS, VANTAGENS E DESVANTAGENS DE UM E DE OUTRO

Conforme a definição disposta na RDC N° 302/2005, Laudo laboratorial é o documento que contém os resultados das análises laboratoriais, validados e autorizados pelo responsável técnico do laboratório ou seu substituto. A sua emissão vem na fase pós-analítica demonstrando o êxito da análise clínica. Devendo ser legível, sem rasuras de transcrição, escrito em língua portuguesa, datado e assinado por profissional de nível superior legalmente habilitado.

O tópico 6.3.3 da mencionada RDC, informa os requisitos mínimos para sua composição, a saber:

a) identificação do laboratório;

b) endereço e telefone do laboratório;

c) identificação do Responsável Técnico (RT);

d) nº. de registro do RT no respectivo conselho de classe profissional;

e) identificação do profissional que liberou o exame;

f) nº. registro do profissional que liberou o exame no respectivo conselho de classe do profissional

g) nº. de registro do Laboratório Clínico no respectivo conselho de classe profissional;

h) nome e registro de identificação do cliente no laboratório;

i) data da coleta da amostra;

j) data de emissão do laudo;

k) nome do exame, tipo de amostra e método analítico;

l) resultado do exame e unidade de medição;

m) valores de referência, limitações técnicas da metodologia e dados para interpretação;

n) observações pertinentes.

Semelhante ao laudo laboratorial há o LAB REPORT e sua emissão também ocorre na fase pós-analítica, com base na Lei Federal do Estados Unidos, 42 U.S.C. §493.1291, o relatório do teste deve indicar:

(1) identificação correta do paciente, o nome do paciente e número de identificação ou um identificador exclusivo do paciente e número de identificação.

(2) O nome e endereço do laboratório, e o local onde o teste foi realizado.

(3) a data do relatório.

(4) o teste realizado.

(5) fonte da amostra, quando apropriado.

(6) O resultado do teste e, se aplicável, as unidades de medida ou interpretação, ou ambos.

(7) Qualquer informação sobre a condição e disposição das amostras que não atendam aos critérios de aceitabilidade do laboratório.

Além de "Intervalos de referência" ou valores "normais" pertinentes, conforme determinado pelo laboratório que realiza os testes, devem estar disponíveis para a pessoa autorizada que solicitou os testes e, se aplicável, o indivíduo responsável pela utilização dos resultados dos testes, a responsabilidade é inerente do Diretor do laboratório, possuindo qualificação mínima para o cargo, previstos na lei.

Cabe aqui uma ressalva fundamental. Embora a RDC 302/2005 traga um conceito de laudo, este não condiz com o conceito determinado nem pelos dicionários brasileiros, tais como o Michaelis que diz expressamente ser o laudo o documento formal em que um especialista emite uma opinião

sobre determinado assunto. Neste ponto, claro está que o documento disponibilizado pelos laboratórios não se trata de laudo e, sim de um relatório de exames que, posteriormente, servirá como arcabouço técnico na emissão de um laudo médico.

## QUAL É O PAPEL DO RT LIBERADOR NOS LABORATÓRIOS?

O Responsável Técnico – RT, conforme nos define a RDC Nº302/2005 é o profissional legalmente habilitado que assume perante a Vigilância Sanitária a Responsabilidade Técnica do laboratório clínico ou do posto de coleta laboratorial.

Dentre suas funções (tópico 5.1.4) está planejar, implementar e garantir a qualidade dos processos, incluindo:

a) A equipe técnica e os recursos necessários para o desempenho de suas atribuições;

b) a proteção das informações confidenciais dos pacientes;

c) a supervisão do pessoal técnico por profissional de nível superior legalmente habilitado durante o seu período de funcionamento;

d) os equipamentos, reagentes, insumos e produtos utilizados para diagnóstico de uso “in vitro”, em conformidade com a legislação vigente;

e) a utilização de técnicas conforme recomendações do fabricante (equipamentos e produtos) ou com base científica comprovada;

f) a rastreabilidade de todos os seus processos.

Salienta-se, por oportuno, que não compõe a responsabilidade do RT Liberador a realização, como parte ativa da análise pós-analítica. A ele recai a responsabilidade de organizar, implementar e supervisionar rotinas laboratoriais e não de realizá-las do ponto de vista prático.

## RESPONSABILIDADE DO LIBERADOR PF E PJ PELO LAUDO

O Responsável Técnico – RT, conforme nos define a RDC Nº302/2005 é o profissional legalmente habilitado que assume perante a Vigilância Sanitária a responsabilidade técnica do laboratório clínico ou do posto de coleta laboratorial. A responsabilidade do RT é sempre subjetiva, decorrente da comprovação de culpa, uma vez que, observados os parâmetros técnicos da execução do trabalho a probabilidade de errar é mínima.

A responsabilidade da pessoa jurídica é objetiva incidindo no presente caso o art. 14 do CDC - Código de Defesa do Consumidor- Lei 8078/1990 sendo independente de culpa, somente excluindo-se a responsabilidade conforme determina o parágrafo terceiro:

> § 3º O fornecedor de serviços só não será responsabilizado quando provar:
> I - que, tendo prestado o serviço, o defeito inexiste;
> II - a culpa exclusiva do consumidor ou de terceiro.

Sendo as únicas excludentes admitidas no nexo de causalidade.

No mesmo sentido, o judiciário reconhece uma relação de consumo, por meio da Apelação Cível Nº1008078-47.2018.8.11.0003 julgado pela Terceira Câmara de Direito Privado, sendo considerado o laboratório uma prestadora de serviços:

> APELAÇÃO CÍVEL – AÇÃO DE INDENIZAÇÃO POR DANOS MORAIS E MATERIAIS – FALHA NA PRESTAÇÃO DE SERVIÇOS – EXAME TOXICOLÓGICO – ERRO NO RESULTADO POSITIVO – NÃO OCORRÊNCIA – RESPONSABILIDADE OBJETIVA – AUSÊNCIA DE REQUISITOS – EXCLUDENTE – SENTENÇA MANTIDA – RECURSO CONHECIDO E DESPROVIDO. A responsabilidade do prestador de serviços pelo "fato do serviço" é regulada pelo art. 14 do Código de Defesa do Consumidor, que prevê a modalidade objetiva de responsabilização, na qual a comprovação da culpa do prestador de serviços é absolutamente desnecessária para que haja a responsabilização da empresa pelos danos causados a seus consumidores, bastando que estes comprovem a prestação de um serviço deficiente, o dano e o nexo de causalidade entre eles. A prestadora dos serviços apenas não responderá pelos danos sofridos pelo consumidor quando apresentar prova cabal de que o defeito alegado não existe ou que o consumidor ou terceiro possuem culpa exclusiva pela ocorrência do dano (§3º do art. 14 do CDC). Comprovado pela prestadora do serviço, por meio de confirmação na contraprova do exame toxicológico, que o primeiro resultado obtido na amostra original não foi equivocado, não há como se imputar a ela o dever de indenizar o autor. A subsunção a novo exame toxicológico 53 (cinquenta e três) dias após a realização do primeiro não é capaz de invalidar. os resultados obtidos no primeiro exame, sobretudo diante de sua

> confirmação pela contraprova. O erro no resultado do exame somente poderia ser cabalmente comprovado por meio da realização de um segundo exame na mesma data, ou pela contraprova. Não tendo o consumidor se desincumbido de seu encargo probatório, à míngua da comprovação da prestação deficiente dos serviços prestados pelo laboratório, não há como acolher seu pedido de reparação por danos morais e materiais. (TJ-MT 10080784720188110003 MT, Relator: DIRCEU DOS SANTOS, Data de Julgamento: 09/12/2020, Terceira Câmara de Direito Privado, Data de Publicação: 22/01/2021)

Num outro julgamento, a 8ª Câmara de Direito Privado do Tribunal de Justiça de São Paulo, nos termos do voto do relator Theodureto Camargo, deu provimento parcial ao recurso interposto pela parte autora, no presente feito, uma gestante.

Considerando que os laboratórios são prestadores de serviços e os pacientes consumidores a Câmara consignou, inicialmente, a incidência do Código de Defesa do Consumidor, na forma do art. 14, caput, da Lei 8.078/90, assentando, nessa linha, que a responsabilidade do laboratório é objetiva.

Destacou, ainda, que a obrigação dos laboratórios é de resultado "[...] já que se trata de atividade especializada na qual se promete o diagnóstico correto" e acostou entendimento jurisprudencial nesse sentido:

"(...) O exame ultrassonográfico para controle de gravidez implica em obrigação de resultado, caracterizada pela responsabilidade objetiva (CDC, art. 14). Precedente do STJ. Defeito do serviço. Não oferecimento ao consumidor da segurança esperada, quanto ao modo de seu fornecimento (CDC, art. 14, § 1º, I). Ocorrência de fortuito interno, que não afasta a imputação. Dano moral. Caracterização (...)" (TJSP, 7ª Câm. Dir. Priv., Ap. 0017447-49.2008.8.26.0348, rel. Des. Luiz Antônio Costa, j. 24.10.2012).

Aqui temos um complicador, porque, embora juridicamente haja a possibilidade de responsabilizar, tanto pessoa física, quanto pessoa jurídica, a utilização de certificados digitais é possível apenas para pessoas físicas, o que mais adiante veremos que também se traduz em grande problemática, pois o processo de análise laboratorial é composto por diversas pessoas e não por uma apenas.

**EM OUTROS PAÍSES, COMO FUNCIONA A ENTREGA DE RESULTADOS DE EXAMES LABORATORIAIS? É AUTOMÁTICA, QUEM ASSINA, UTILIZA-SE ALGUM TIPO DE CERTIFICAÇÃO DIGITAL?**

A referência utilizada aqui foi a Lei Federal do Estados Unidos prevista no 42 U.S.C, §493.1359, na qual constam as atribuições do Diretor, indicando suas funções gerais, entre elas a operação e administração do laboratório, incluindo a comunicação imediata, precisa e proficiente dos resultados dos testes, ainda no §493.1291, g, o laboratório deve imediatamente reportar à entidade que solicitou os testes, ou ao responsável que solicitou o teste, em caso de o resultado indicar uma condição de ameaça iminente a vida ou valores muito fora dos padrões.

Quanto a um processo de certificação digital, o que determina a lei é a existência de um processo de autenticação próprio do laboratório, para que identifique um determinado paciente, descrito na §493.1291, i.

Em outras palavras, a legislação norte-americana também não dispõe da obrigatoriedade da utilização de certificados digitais, ao contrário, assim como o faz em diversos outros setores, delega ao laboratório o processo de autenticação, salvaguardados todos os riscos a que não só o laboratório, bem como seu responsável técnico estão expostos.

## COMO ACONTECE A LIBERAÇÃO AUTOMÁTICA DE RESULTADOS DE EXAMES EM LABORATÓRIOS DE ANÁLISES CLÍNICAS?

- São inseridos parâmetros no sistema de informação laboratorial (SIL), com base em critérios estabelecidos pelo laboratório, sendo possível adaptar e personalizar conforme necessário.
- São inseridos parâmetros no sistema de informação laboratorial (SIL), com base em critérios estabelecidos pelo laboratório, sendo possível adaptar e personalizar conforme necessário.
- A auto verificação uniformiza os critérios de liberação e melhora a eficiência no processo, garantindo a segurança do paciente.
- A liberação automática dos resultados ocorre diretamente do equipamento sem a intervenção humana.
- Facilitando o contato com o cliente por meio eletrônico, além de promover de maneira eficiente e sustentável a organização dos fluxos laboratoriais.

## PROCESSO PÓS-ANALÍTICO E O IMPACTO DA ALTERAÇÃO DAS FASES

O processo de produção de resultados de exames de análises clínicas pode ser resumidamente descrito no diagrama abaixo:

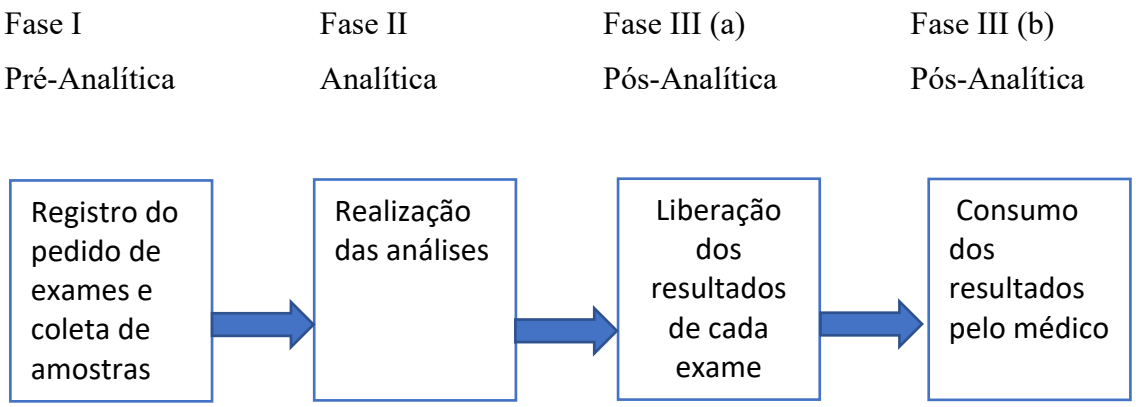


Figura 1 – Fluxo Simplificado Laboratorial

A figura apresenta um fluxograma básico dos processos de entrada, processamento de exames laboratoriais, liberação consumo dos resultados em um ambiente laboratorial. A simples análise da Figura 1 evidencia uma importante imprecisão existente na redação da RDC 302 e que todos os envolvidos na produção de Sistemas de Informação Laboratorial (SIL) esperavam ver resolvida em uma nova redação: o equivocado entendimento de que os resultados laboratoriais são consolidados em um documento único – o chamado "laudo" – e que esse documento é "assinado" por um profissional habilitado.

Para a correta compreensão da citada imprecisão, ao menos quatro aspectos devem ser analisados:

- os exames laboratoriais solicitados pelo médico em um pedido de exames são produzidos no laboratório de análises clínicas separadamente em distintos departamentos e com o envolvimento de diferentes profissionais;
- cada exame leva um tempo distinto dos demais para ficar pronto, variando de poucos minutos a diversos dias, e é interesse do paciente e da equipe que o assiste que os resultados sejam liberados individualmente e assim que disponíveis;
- os resultados de exames laboratoriais são consumidos em múltiplas mídias e em múltiplas apresentações, variando em função do contexto do consumo, cabendo exclusivamente ao médico e ao paciente definirem quando e como consumirão tais resultados, sendo inúmeras vezes absolutamente inadequada a consolidação de todos os resultados em um único documento;

- a grande maioria dos laboratórios de análises clínicas brasileiros produz de alguns milhares a vários milhões de exames laboratoriais mensalmente, o que só é possível graças à adoção de algoritmos de liberação automatizada de resultados, pois seria inviável a análise pessoal de cada resultado frente aos altos volumes demandados.

Em todo laboratório de análises clínicas, uma vez recebido o pedido de exames, o SIL encarrega-se de calcular quantas e quais amostras de material biológico devem ser obtidas do paciente para a realização das análises. As amostras são então colhidas do paciente e tomam cada qual o seu próprio rumo dentro do laboratório: amostras de urina seguem para a Urinálise; amostras de soro, para a Bioquímica, para a Endocrinologia, para a Imunologia ou ainda para outras especialidades; amostras de fezes, para a Parasitologia; sem citar a Biologia Molecular, a Microbiologia etc. Em cada setor são realizados os exames do pedido médico a ele competentes, sendo processados em equipamentos específicos para cada análise e acompanhados igualmente por profissionais específicos. Portanto, é notória a imperfeição da redação que espera que um único "profissional habilitado" assine o conjunto de todos os resultados produzidos pelo laboratório para aquele pedido médico, já que o conjunto dos resultados é uma obra multidisciplinar e produzida com o envolvimento de distintas tecnologias e distintos profissionais, cada qual respondendo obviamente pelos resultados que produzira.

O segundo ponto a se observar é que os exames demandam tempos distintos para terem seus resultados produzidos e liberados. Desde um Tempo de Sangramento, que se resolve em minutos, passando por exames de triagem como o Hemograma ou a Urina Tipo I, disponíveis em até um dia útil, e chegando a exames como a Cultura de Microbactérias, que chegam a consumir 45 dias para ficarem prontos, os exames de análises clínicas são realizados em metodologias distintas e com distintos graus de automação e complexidade, levando consequentemente à disponibilização dos resultados em tempos também diferentes. A parte mais importante dessa característica do processo é que pacientes e médicos não esperam pela conclusão de todos os exames para tomarem as condutas necessárias. Ao contrário, exames mais rápidos e mais imprecisos fundamentam o início do tratamento, que pode ser corrigido ou complementado à medida que os resultados dos exames mais complexos e específicos ficam prontos. Disponibilizar os resultados individual e progressivamente não é uma decisão do laboratório: é uma demanda legítima dos solicitantes, incompatível com a equivocada expectativa de que os resultados devam compor um "laudo" a ser assinado antes de ser consumido por quem deles depende, pois isso só seria viável após a conclusão da última análise.

O terceiro ponto decorre dos anteriores e os complementa. Para o paciente em emergência, a disponibilidade de um resultado de exame laboratorial pode significar a diferença entre viver e

morrer. Qualquer serviço de saúde privilegiará a informação rápida e objetiva e não a presença de uma assinatura. São frequente e igualmente objetivas as necessidades durante intervenções cirúrgicas e durante o acompanhamento de internados em UTIs, a ponto de provocar a constante evolução dos equipamentos de beira de leito, que produzem os resultados necessários à orientação da conduta sem a necessidade de trânsito das amostras até os laboratórios – muitas vezes antes mesmo que o pedido médico seja introduzido no SIL! Nessas ocasiões, frequentemente se dispensam inclusive outros elementos exigidos de um relatório final, como os valores de referência, por exemplo, tamanha a objetividade e urgência esperadas. Mais uma vez, exigir que um único documento será consolidado com todos os resultados, e assinado uma vez produzido, é negar provimento de solução a casos como os citados e a tantos outros. Que se frise mais uma vez: os resultados de exames não são entregues aos interessados; antes, são consumidos por eles, quando e como se fizerem necessários.

Por fim, resta lembrar que os volumes de exames de análises clínicas produzidos pelos laboratórios brasileiros são muito grandes, existindo serviços que produzem milhões de resultados mensalmente, além de uma infinidade na casa dos milhares no mesmo período. Existe longa e madura prática de liberação automatizada de resultados de exames através da adoção de algoritmos no SIL capazes de realizar as mesmas considerações de um profissional habilitado em circunstâncias específicas que envolvem, além dos resultados propriamente dito, características pessoais como sexo e idade, resultados anteriores e outras informações.

A maior parte da produção laboratorial é frequentemente liberada dessa forma, sendo retidos para análise individualizada aqueles resultados que não se enquadrem nos critérios de liberação automatizada previamente configurados. Estes mecanismos são os responsáveis por fazer chegar ao paciente e ao médico os resultados no menor tempo possível, praticamente limitando esse tempo aos processos pré-analíticos e analíticos, reduzindo a liberação dos resultados de grande parte da produção laboratorial a poucos segundos. Esta é a realidade da imensa maioria dos exames de análises clínicas produzidos no Brasil e no mundo, e aponta, igualmente às anteriores considerações, à inexistência de documentos únicos, consolidados e assinados, os chamados "laudos" pela RDC 302 e pela presente CP 912. Em seu lugar, mais uma vez, resta clara a existência de resultados individuais, liberados isoladamente e, em grande maioria, de maneira automática.

Merece consideração de que a redação original da RDC 302, mantida na CP 912 quando se refere ao "laudo", é vítima de anacronismo. À época em que fora publicada (2005), a grande via de disseminação dos resultados laboratoriais era o papel, já que os recursos digitais, hoje tão onipresentes, ainda engatinhavam. Como o papel requer deslocamento do interessado ao laboratório ou o envio do laboratório ao paciente, era prática corrente "juntar" os resultados antes da "entrega"

ao interessado, seja o próprio paciente, seja o médico que o assiste, no sentido de minimizar os custos dessa logística. A esses resultados juntados, historicamente se convencionou chamar de "laudo", por aquele relatório guardar alguma semelhança com o resultado do trabalho de um perito, mas que guarda diferenças importantes que inviabilizam a permanência desta terminologia, como mais à frente demonstraremos. Mesmo àquela época, em ambiente de emergência e de urgência, como nos hospitais, era prática comum que os resultados fossem passados aos médicos pelo telefone ou já nos primeiros terminais de consulta dos SIL, pois não convinha aguardar a chegada do papel. Também àquela época já eram verdadeiras todas as considerações sobre os diferentes tempos necessários à realização dos exames, sobre os distintos caminhos seguidos por cada amostra dentro do laboratório, já eram muito grandes os volumes da produção laboratorial e já se adotavam algoritmos de liberação automatizada, mas ainda padecíamos dos meios para a disponibilização imediata aos interessados. Acontece que a digitalização da sociedade trouxe a vantagem inafastável do acesso à informação rápida e objetivamente. Assim como cotidianamente matamos a curiosidade sobre um termo qualquer em uma pesquisa no Google, com muito mais mérito médicos e cidadãos consomem seus resultados de análises clínicas tão logo encontram-se disponíveis. Em suma, não são mais produzidos laudos e, muito menos, entregues esses laudos aos interessados; os resultados de cada exame são liberados individualmente, para consumo por iniciativa dos interessados, quando e como lhes seja mais conveniente. Essa sutil diferença, aliada às considerações anteriores, demonstra claramente: 1) a inexistência de um documento a ser assinado, e 2) a impossibilidade de se lhe aplicar uma assinatura.

Além da questão da efetividade do SADT-TAC existe a questão do gerenciamento dos certificados digitais de assinatura da ICP-Brasil que deverão ser alocados a cada profissional habilitado dos SADT-TAC ou a seus responsáveis. Isso impactará de maneira significativa os procedimentos atuais e, em caso de algum contencioso, poderá ser colocada em dúvida a correta manipulação dos certificados digitais. Isso mais uma vez sem levar-se em consideração o total desconhecimento das boas práticas de distribuição e gestão dos certificados, tendo em vista que um único SADT-TAC pode chegar a ter vários profissionais habilitados em localidades diversas o que implicará em um grande volume de certificados digitais que deverão ser custodiados. Ressalva-se neste ponto, a ausência de uma definição clara de quem deva ser este profissional habilitado o que, muito embora num primeiro momento possa não indicar problemas, na prática faz toda a diferença no estabelecimento das responsabilidades, inclusive do ponto de vista da formação deste profissional para a ocupação de uma função eminentemente técnica.

Outra questão que se mostra preocupante, é o prazo de validade dos certificados digitais e o gerenciamento da validade de cada certificado alocada aos diversos SADT-TAC. Isto implicará num

aumento dos custos operacionais e até mesmo uma responsabilização em caso de mal-uso dos mesmos, sem levar em consideração a possibilidade da necessidade de acompanhamento de certificados revogados.

Por todo o exposto, apesar de a certificação digital constituir um processo utilizado para a garantia da integridade, autenticidade e o não-repúdio, nos casos dos SADT-TAC o emprego da certificação digital irá impactar negativamente na efetividade dos processos realizados pelos SADT-TACs, o que torna a sua inserção nos processos acima descritos inviável.

## A Lei 14.063/2020 e a disposição sobre o uso de assinaturas eletrônicas em questões de saúde

Para além de todo o exposto nos itens anteriores quanto à inviabilidade material e factual da utilização de certificados digitais em resultados de exames laboratoriais, dispõe a Lei 14.043/2020 sobre os diferentes tipos de assinaturas em documentos que tratam de questões de saúde.

Em seu art. 4º, I e II, fica evidente a possibilidade de utilização da assinatura em sua forma avançada, independentemente da utilização de certificado provido pela ICP-Brasil como a única forma confiável e fidedigna para a assinatura eletrônica de documentos em saúde, tal como expresso abaixo:

Para efeitos desta Lei, as assinaturas eletrônicas são classificadas em:

I - assinatura eletrônica simples:

a) a que permite identificar o seu signatário;

b) a que anexa ou associa dados a outros dados em formato eletrônico do signatário;

**II - assinatura eletrônica avançada: a que utiliza certificados não emitidos pela ICP-Brasil ou outro meio de comprovação da autoria e da integridade de documentos em forma eletrônica, desde que admitido pelas partes como válido ou aceito pela pessoa a quem for oposto o documento, com as seguintes características:**

**a) está associada ao signatário de maneira unívoca;**

**b) utiliza dados para a criação de assinatura eletrônica cujo signatário pode, com elevado nível de confiança, operar sob o seu controle exclusivo;**

**c) está relacionada aos dados a ela associados de tal modo que qualquer modificação posterior é detectável;**

Em última análise, a Lei 14.063/2020 contrapõe a Consulta Pública 912, vez que ao admitir a possibilidade de outros tipos de assinatura, pela hierarquia das normas, se sobrepõe ao instrumento regulatório.

Ademais, corroboram a posição acima defendida os arts. 13 e 14, I, II do mesmo diploma legal, *in verbis*:

Art. 13. Os receituários de medicamentos sujeitos a controle especial e os atestados médicos em meio eletrônico, previstos em ato do Ministério da Saúde, somente serão válidos quando subscritos com assinatura eletrônica qualificada do profissional de saúde.
Parágrafo único. As exigências de nível mínimo de assinatura eletrônica previstas no caput deste artigo e no art. 14 desta Lei não se aplicam aos atos internos do ambiente hospitalar.

Art. 14. Com exceção do disposto no art. 13 desta Lei, os documentos eletrônicos subscritos por profissionais de saúde e relacionados à sua área de atuação são válidos para todos os fins quando assinados por meio de:
I - assinatura eletrônica avançada; ou
II - assinatura eletrônica qualificada.

## PROPOSIÇÃO TÉCNICA ALTERNATIVA

### O QUE É LAUDO LABORATORIAL?

Conforme nos disponibiliza essa definição a RDC N° 302/2005, Laudo laboratorial é o documento que contém os resultados das análises laboratoriais, validados e autorizados pelo responsável técnico do laboratório ou seu substituto. A sua emissão vem na fase pós-analítica demonstrando o êxito da análise clínica. Devendo ser legível, sem rasuras de transcrição, escrito em língua portuguesa, datado e assinado por profissional de nível superior legalmente habilitado. No tópico 6.3.3, da mencionada RDC, nos informa os requisitos mínimos para sua composição sendo elas:

a) identificação do laboratório;
b) endereço e telefone do laboratório;
c) identificação do Responsável Técnico (RT);
d) Nº. de registro do RT no respectivo conselho de classe profissional;
e) identificação do profissional que liberou o exame;

f) Nº. registro do profissional que liberou o exame no respectivo conselho de classe do profissional;
g) Nº. de registro do Laboratório Clínico no respectivo conselho de classe profissional;
h) nome e registro de identificação do cliente no laboratório;
i) data da coleta da amostra;
j) data de emissão do laudo;
k) nome do exame, tipo de amostra e método analítico;
l) resultado do exame e unidade de medição;
m) valores de referência, limitações técnicas da metodologia e dados para interpretação;
n) observações pertinentes.

Para fins deste parecer e para padronização, cada um dos campos, ou conjunto de campos, que compõem o laudo laboratorial serão aqui chamados de "mensagem". Portanto, uma "mensagem" é um campo de dados para o qual se pretende garantir integridade e autenticidade, indo desde um campo de dado de informações pessoais, até os dados de resultados de um exame, passando por dados de identificação de exames, e dados administrativos, entre outros.

**QUAL A DIFERANÇA ENTRE INTEGRIDADE E AUTENTICIDADE**

Existem 3 pilares principais da segurança da informação: a integridade, a autenticidade e a confidencialidade. Para fins deste parecer estamos interessados em discutir os mecanismos de INTEGRIDADE e AUTENTICIDADE.

Segundo nos ensina Schneier (1995) a INTEGRIDADE deve garantir as características originais dos dados, garantindo que eles se mantenham íntegros e livres de alterações que possam ser produzidas por pessoas ou processos não autorizados. Assim, a integridade deve garantir que os dados apresentados correspondem ao que se espera, e que estes não tenham sido alterados ou corrompidos por qualquer meio. A integridade de uma mensagem garante que aquele dado não foi alterado sem autorização. Havendo qualquer modificação indevida nos dados, significa que houve perda da integridade, portanto se faz necessária a implementação de mecanismos de controle, com o intuito de impedir a alteração não autorizada das informações.

Ainda em seu tratado, Schneier considera que a AUTENTICIDADE é uma condição que garante que um dado é legítimo e que ele foi produzido ou alterado por aquele que tem a devida autorização para tal. Assim, a autenticidade deve garantir que os dados apresentados são provenientes de quem o diz ser, ou de quem tem os direitos para tal. A autenticidade deve ser capaz de assegurar que uma mensagem foi emitida legitimamente porque tem o direito de fazê-lo. Havendo qualquer dúvida quanto à origem dos dados, significa que houve perda da autenticidade, portanto se faz necessária a implementação de mecanismos de controle, com o intuito de impedir a criação e alteração não autenticada das informações.

**TÉCNICA DE RESUMO DE MENSAGEM "HASH" PARA INTEGRIDADE**

Segundo Menezes et al. (1996) a técnica de Resumo de Mensagem, ou "hash", é um método de validação de integridade que gera um código único para ser usado como um valor de segurança de uma mensagem ou bloco de dados. Uma função hash recebe uma mensagem de comprimento variável e produz uma mensagem resumida, de comprimento fixo, como sua saída, independentemente do tamanho da entrada.

Uma função hash ideal tem as seguintes propriedades: É rápida para executar; pode retornar uma gama enorme de valores de hash; gera um hash único para cada entrada única (sem colisões); gera valores de hash diferentes para valores de entrada semelhantes; e valores de hash gerados não têm um padrão discernível em sua distribuição. Não existe função de hash ideal, mas cada uma tem como objetivo operar o mais próximo possível do ideal. De acordo com [Paar, 2014] as funções de hash criptográficas vêm com três requisitos adicionais em relação às funções de hash normais.

O primeiro requisito é que uma função hash criptográfica seja unilateral. Isso significa que, dado um resumo, deve ser computacionalmente inviável inverter a função hash e calcular sua pré-imagem.

Por exemplo, dado o hash "88ebe8bf78169fcfe6d8a68b040ca600284bacd3", não deveria haver maneiras de descobrir que a pré-imagem era "TEXTO", a não ser por tentativas potenciais de força bruta. Isso não é tecnicamente verdadeiro para todas as funções de hash comuns (não criptográficas) como, por exemplo, somas de verificação do tipo CRC32 .

O segundo requisito é que uma função hash criptográfica exiba um “efeito de avalanche”. O efeito de avalanche basicamente diz que se qualquer bit for alterado na pré-imagem, isso deve desencadear uma "avalanche" que embaralha todos os outros bits. Assim, duas entradas muito semelhantes, que diferem em apenas um bit, não devem ter nenhuma relação discernível entre suas saídas.

A última propriedade que uma função hash criptográfica precisa ter é a resistência à colisão. Quando duas entradas têm hash para a mesma saída, isso é conhecido como colisão, e é algo que deve ser evitado.

Ainda com relação aos algoritmos com funções hash criptográficas unilaterais, existe a família SHA (Secure Hash Algorithm). A primeira versão do algoritmo foi o SHA-1, e mais tarde foi seguida pelo SHA-2.

A família de funções “MD” são funções de hash criptográficas unilaterais que englobam uma sequência de dígitos para proteger a integridade de dados. Explicando resumidamente sua evolução: o algoritmo MD original (chamado MD1) foi seguido por uma versão modificada MD2, mas ambos foram considerados bastante fracos, sendo precedidos e atualizados pelo MD3, o qual acabou não sendo lançado, sendo que o MD4, que também apresentou problemas, mas forneceu as bases teóricas para MD5 e SHA.

Conforme discutido por Aggarwal et al (2014), a função MD5 [RFC-1321, 1992] produz resumos de mensagem de 128 bits a partir de mensagens de entrada de comprimento qualquer. Ela foi projetada para uso em criptografia e assinaturas digitais, mas também foram descobertas vulnerabilidades ao longo do tempo, portanto, não é mais recomendada para esse propósito. No entanto, ela ainda é usada para particionamento de bancos de dados e somas de verificação de computação para validar as transferências de arquivos.

Enquanto MD5 produz um hash de 128 bits, o SHA-1, por sua vez, gera um hash de 160 bits (20 bytes - no formato hexadecimal, o que é um número inteiro de 40 dígitos). Assim como o MD5, ele foi projetado para aplicativos de criptografia, mas logo descobriu-se que também possuía vulnerabilidades. Atualmente ele não é mais considerado mais resistente a ataques do que o MD5.

Ou seja, a função SHA-1, de forma similar à MD5, também não deve ser usada para os fins dos quais se tratam este documento.

Entre os algoritmos com funções hash criptográficas unilaterais, o mais recomendado é o SHA-256, o qual é endossado pelo Instituto Americano de Padrões e Tecnologia (USA/NIST). O algoritmo SHA-256 retorna um valor hash de 256 bits ou 64 dígitos hexadecimais. Embora não seja totalmente perfeito, a pesquisa atual indica que é consideravelmente mais seguro.

Mesmo para uma função hash como SHA-256, que tem um intervalo de tamanho 2^256 ,se for possível tentar (2^256) + 1 combinações de números, terá que se obter pelo menos uma colisão. Portanto, sabemos que todas as funções hash devem ter colisões – uma vez que todas as funções hash têm tamanhos de saída finitos, mas com infinitas entradas possíveis. Mas, para todos os efeitos práticos, 2^256 é tão grande que é efetivamente inumerável. Para efeito de comparação, o número de átomos no universo é aproximadamente 2ˆ260. Portanto, embora saibamos que devem existir colisões, nunca encontraremos uma durante a vida do universo e, de fato, nunca foi encontrada uma colisão para o SHA-256. A fronteira da tratabilidade computacional atual é de aproximadamente 2 ^ 80. Abaixo disso, é plausível que um ator poderoso o suficiente possa usar força bruta para quebrar esta criptografia. Adicione algum espaço extra para algumas décadas de proteção futura, e isso leva a segurança de até 128 bits como uma boa regra prática. Então, se quisermos aumentar o tamanho do intervalo, pode-se aumentar o nível de nossa função hash.

## TÉCNICA *HMAC* PARA VERIFICAÇÃO DE AUTENTICIDADE

As funções de hash são altamente versáteis, mas frequentemente seu uso é recomendado apenas para **verificação de integridade**. Quando **existe necessidade de verificação de integridade E autenticidade**, é recomendável a utilização de um algoritmo de HMAC ou *Hash-based Message Authentication Code* [RFC-2104, 1997]. O *Message Authentication Code* (ou Código de Autenticação de Mensagem) trata-se de uma função matemática ou algoritmo que fornece verificação de integridade e autenticidade utilizando uma chave secreta. Trata-se de uma técnica proposta em 1996 por Bellare, Canetti e Krawczyk, e que se tornou um padrão na Internet [BCK, 1996].

Em criptografia, um código HMAC ou *Hash-based Message Authentication Code* (Código de Autenticação Baseado em *Hash*) é um mecanismo que fornece a verificação de integridade e

autenticidade de dados, com base em uma chave secreta associada a uma função de hash criptográfica. Enquanto a função de hash é usada para **verificar a integridade dos dados**, uma chave secreta é utilizada para **verificação dos valores de autenticação da mensagem**.

Qualquer função hash criptográfica, como SHA-256 ou SHA-3, pode ser usada no cálculo de um HMAC. O algoritmo MAC resultante é denominado HMAC-X, em que X é a função de hash usada (por exemplo, HMAC-SHA256). A força criptográfica do HMAC depende da força criptográfica da função de *hash* subjacente, do tamanho de sua saída de hash e do tamanho e qualidade da chave. A mais indicada é a função SHA-256. Outra propriedade interessante é a possibilidade de substituição da função hash subjacente quando ela se tornar obsoleta ou quando surgirem melhores opções.

O HMAC é um algoritmo e esse cálculo é simplesmente a execução desse algoritmo. A grosso modo, a função HMAC é definida por:

$$HMAC(K, m) = hash(K1 + hash(K2 + m))$$

onde:

- ***K :*** é a chave secreta.
- ***M*****:** é a mensagem ou dados sendo autenticados e verificados.
- ***hash*** **:** é a função de hash escolhida (sha1, sha256, etc).
- ***K1*** e ***K2 :*** são chaves secretas derivadas da chave original ***K***.
- **+** ***:*** é a operação de concatenação de strings.

A **Figura 1** representa a aplicação de um Código de autenticação de mensagem.

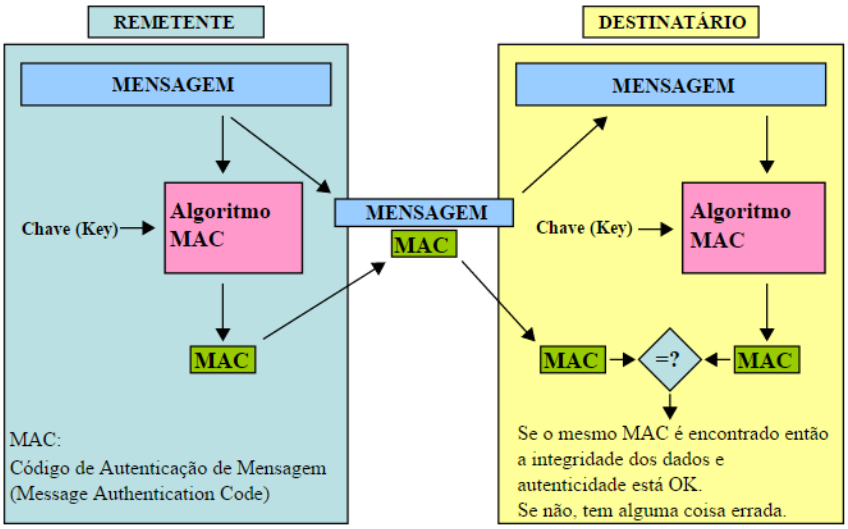


**Figura 1** - Representação de troca de informações usando código de autenticação de mensagem (MAC). Adaptada de: *Message Authentication Code – Wikipedia* [Wiki, 2021]

Conforme resumidamente discutido em [Wiki, 2022]: "O HMAC usa duas passagens de computação de hash. A chave secreta é usada primeiro para derivar duas chaves – interna e externa. A primeira passagem do algoritmo produz um hash interno derivado da mensagem e da chave interna. A segunda passagem produz o código HMAC final derivado do resultado do hash interno e da chave externa. Assim, o algoritmo fornece melhor imunidade contra os ataques de extensão de comprimento. Uma função de hash iterativa divide uma mensagem em blocos de tamanho fixo e itera sobre eles com uma função de compactação. Por exemplo, SHA-256 opera em blocos de 512 bits. O tamanho da saída do HMAC é o mesmo da função hash subjacente (por exemplo, 256 e 512 bits no caso de SHA-256 e SHA3-512, respectivamente), embora possa ser truncado, se desejado. O HMAC não criptografa a mensagem. Em vez disso, a mensagem (criptografada ou não) deve ser enviada juntamente com o hash HMAC. As partes com a chave secreta farão o hash da mensagem novamente e, se for autêntica, os hashes recebidos e calculados serão correspondentes."

Em seguida um código Python simplificado apresenta o uso de uma função HMAC-SHA256:

```
#!/usr/bin/python
from hashlib import sha256
from hmac import new as hmac
key = "aaI08Vcaxpqr0PQAbmasetkletpaxzUoP1BP1oASnb"
message = "meu_resultado"
print "%s" % hmac(key, message, sha256).digest().encode('base64')[:-1]
```

Alguns exemplos de programação e uso HMAC são apresentadas a seguir:

Em seguida um código Python simplificado apresenta o uso de uma função HMAC-SHA256:

```
#!/usr/bin/python
from hashlib import sha256
from hmac import new as hmac
key = "aaI08Vcaxpqr0PQAbmasetkletpaxzUoP1BP1oASnb"
message = "meu_resultado"

print "%s" % hmac(key, message, sha256).digest().encode('base64')[:-1]
```

Como exemplo de aplicação de uso HMAC são apresentadas, a seguir, algumas funções em Python e em C#. Apesar dos exemplos serem apresentados nessas linguagens, por opção desta serem linguagens algorítmicas de fácil entendimento, estes exemplos podem ser extrapolados facilmente para outras bibliotecas, funções e linguagens.

### A. Exemplos de funções HMAC em Python V.3.10

A aplicação destas funções depende do Módulo "***hashlib***" do Python [HASHLIB, 2022]. Este módulo implementa uma interface comum para muitos algoritmos de hash seguro e resumo de mensagem. Estão incluídos os algoritmos de hash seguros FIPS SHA1, SHA224, SHA256, SHA384 e SHA512, bem como o algoritmo MD5 da RSA (definido em Internet [RFC-1321, 1992]). Os termos "hash seguro" e "resumo da mensagem" são intercambiáveis. Algoritmos mais antigos eram chamados de resumos de mensagens. O termo moderno é hash seguro.

O módulo ***hmac*** [HMAC, 2022] implementa o algoritmo HMAC conforme descrito pelo RFC 2104 [RFC-2104, 1997]. Alguns exemplos de utilização estão a seguir.

- hmac.**new**(*key*, *msg=None*, *digestmod=''*)

  Retorna um novo objeto "***hmac***". O valor "***key***" é um objeto do tipo *bytes* ou *bytearray* onde se fornece a chave secreta. Se "***msg***" estiver presente, a atualização da chamada do método update(msg) será executada. O "***digestmod***" é o nome da função ou algoritmo de hash a se utilizado. Obsoleto desde a versão 3.4, removido na versão 3.8: MD5 como compilação padrão implícita para "***digestmod***" está obsoleto. O parâmetro "***digestmod***" agora é necessário.

- hmac.**digest**(*key*, *msg*, *digest*)

  Retorna o resumo da msg para a chave secreta "***key***" e o resumo "***digest***" fornecidos. A função é equivalente a HMAC(key, msg, digest).digest(), mas usa uma implementação C otimizada ou embutida, a qual é mais rápida para mensagens que cabem na memória. Os parâmetros ***key***, ***msg*** e ***digest*** têm o mesmo significado que em ***new()***.

Um objeto HMAC apresenta os seguintes métodos:

- HMAC.**update**(*msg*)

  Atualiza o objeto ***hmac*** com ***msg***. Chamadas repetidas são equivalentes a uma única chamada com a concatenação de todos os argumentos: m.update(a); m.update(b) é equivalente a m.update(a + b).

- HMAC.**digest**()

  Retorna o resumo ***digest*** dos bytes passados para o método ***update()*** até o momento. Este objeto de bytes terá o mesmo comprimento que ***digest_size*** do resumo fornecido ao construtor. Ele pode conter bytes não ASCII, incluindo bytes NUL.

- HMAC.**hexdigest**()

  Assim como ***digest()*** , exceto que o ***digest*** é retornado como uma string com o dobro do comprimento contendo apenas dígitos hexadecimais. Isso pode ser usado para enviar o valor com segurança em e-mail ou outros ambientes não binários.

- HMAC.**copy**()

  Retorne uma cópia (“clone”) do objeto ***hmac***. Isso pode ser usado para calcular com eficiência os resumos de strings que compartilham uma substring inicial comum.

Um objeto hash tem os seguintes atributos:

- HMAC.**digest_size**

  O tamanho do resumo ***HMAC*** resultante em bytes.

- HMAC.**block_size**

  O tamanho do bloco interno do algoritmo hash em bytes.

- HMAC.**name**

  O nome canônico deste ***HMAC***, sempre em letras minúsculas, por exemplo, hmac-sha256.

Este módulo também fornece a seguinte função auxiliar:

- hmac.**compare_digest**(*a*, *b*)

  Retorna a == b. Esta função usa uma abordagem projetada para prevenir a análise de tempo, evitando comportamento de curto-circuito baseado em conteúdo, tornando-a apropriada para criptografia. Os valores ***a*** e ***b*** devem ser do mesmo tipo: str (somente ASCII, como, por exemplo, retornado por *HMAC.hexdigest ()* ) ou um objeto semelhante a bytes.

## B. Exemplo de código usando HMAC-SHA-256 em Python[1].

Este é um exemplo de um trecho de Código Python V.3.10 para uso de HMAC em uma mensagem ou bloco de dados. Note que, como o HMAC acompanha a mensagem assinada em todo o tempo da troca da mensagem, o fato de HMAC ser truncado não é significante, pois pode ser conferido a qualquer momento caso o receptor da mensagem tenha a mesma length, ikm, salt e info passadas como parâmetro de hkdf(), que também deve ser conhecido pelo lado receptor. Apesar do exemplo ser apresentado em Python, por opção se ser uma linguagem algorítmica de fácil entendimento, estes exemplos podem ser extrapolados facilmente para outras bibliotecas, funções e linguagens.

```
# Comentários
# Créditos: Marcelo Shiniti Uchimura
# https://pt.stackoverflow.com/questions/305854/codigo-em-python-usando-biblioteca-hmac
#

import hashlib
import hmac
from math import ceil

# As palavras vão ter 32 bytes.
hash_len = 32

# Define um método que aplica SHA256 com a key e os data informados.
def hmac_sha256(key, data):
    return hmac.new(key, data, hashlib.sha256).digest()

# Cria um HMAC do tamanho length, com a chave ikm,
# o sal salt é uma informação, também usada para a criação do HMAC, info.
def hkdf(length, ikm, salt=b"", info=b""):

    # Gera uma chave preliminar com o salt e o ikm informados.
    prk = hmac_sha256(salt, ikm)
```

[1] Fonte: https://pt.stackoverflow.com/questions/305854/codigo-em-python-usando-biblioteca-hmac (07/01/2022)

```
    # Declara um espaçador t.
    t = b""

    # Declara o HMAC final okm.
    okm = b""

    # Laço para gerar o HMAC; itera-se sobre um vetor [0..ceil(length / 32)]
    for i in range(ceil(length / hash_len)):
        # O espaçador guarda o resultado do algoritmo de hashing
        # a cada iteração, usando como base para tal a si próprio,
        # a info e um vetor de zeros de tamanho variável, apenas
        # como preenchedor (para que t não fique viciado).
        t = hmac_sha256(prk, t + info + bytes([1+i]))

        # Concatena t em okm
        okm += t

    # Recorta okm do tamanho length informado.
    return okm[:length]
```

### C. Exemplo de código Java[2]

O exemplo de código Java a seguir mostra como produzir um HMAC usando as funções padrão da API de segurança Java:

```
import javax.crypto.Mac;
import javax.crypto.spec.SecretKeySpec;
import java.util.Formatter;


public class Main {


  public static String toHexString(byte[] bytes) {
    Formatter formatter = new Formatter();
    for (byte b : bytes) {
      formatter.format("%02x", b);
```

[2] Fonte: https://steelmon.wordpress.com/2015/12/11/the-wonderful-applications-of-hmac/ (07/01/2022)

```
    }
    return formatter.toString();
  }

  public static void main(String[] args) throws Exception {
    byte[] message = "Message to be processed".getBytes("UTF8");
    byte[] keybytes = "PoorKey".getBytes("UTF8");
    SecretKeySpec key = new SecretKeySpec(keybytes,"HmacSHA1");
    Mac hmac = Mac.getInstance("HmacSHA1");
    hmac.init(key);
    byte[] bytes = hmac.doFinal(message);
    System.out.println("HMAC: " + toHexString(bytes));
  }
}
```

D. Exemplo de aplicação código usando HMAC-SHA-256 em Linguagem C#[3] para verificar a integridade e autenticidade de um arquivo.

```
// O exemplo a seguir mostra como assinar um arquivo usando
// o HMACSHA256 objeto e, em seguida, como verificar o arquivo.

using System;
using System.IO;
using System.Security.Cryptography;

public class HMACSHA256example
{

    public static void Main(string[] Fileargs)
    {
        string dataFile;
        string signedFile;
```

[3] Fonte: https://docs.microsoft.com/pt-br/dotnet/api/system.security.cryptography.hmacsha256 (07/01/2022)

```
    //If no file names are specified, create them.
    if (Fileargs.Length < 2)
    {
        dataFile = @"text.txt";
        signedFile = "signedFile.enc";

        if (!File.Exists(dataFile))
        {
            // Create a file to write to.
            using (StreamWriter sw = File.CreateText(dataFile))
            {
                sw.WriteLine("Here is a message to sign");
            }
        }
    }
    else
    {
        dataFile = Fileargs[0];
        signedFile = Fileargs[1];
    }
    try
    {
        // Create a random key using a random number generator. This would be the
        //  secret key shared by sender and receiver.
        byte[] secretkey = new Byte[64];
        //RNGCryptoServiceProvider is an implementation of a random number generator.
        using (RNGCryptoServiceProvider rng = new RNGCryptoServiceProvider())
        {
            // The array is now filled with cryptographically strong random bytes.
            rng.GetBytes(secretkey);

            // Use the secret key to sign the message file.
            SignFile(secretkey, dataFile, signedFile);

            // Verify the signed file
            VerifyFile(secretkey, signedFile);
        }
    }
    catch (IOException e)
    {
        Console.WriteLine("Error: File not found", e);
    }
} //end main
```

```
// Computes a keyed hash for a source file and creates a target file with the keyed hash
// prepended to the contents of the source file.
public static void SignFile(byte[] key, String sourceFile, String destFile)
{
    // Initialize the keyed hash object.
    using (HMACSHA256 hmac = new HMACSHA256(key))
    {
        using (FileStream inStream = new FileStream(sourceFile, FileMode.Open))
        {
            using (FileStream outStream = new FileStream(destFile, FileMode.Create))
            {
                // Compute the hash of the input file.
                byte[] hashValue = hmac.ComputeHash(inStream);
                // Reset inStream to the beginning of the file.
                inStream.Position = 0;
                // Write the computed hash value to the output file.
                outStream.Write(hashValue, 0, hashValue.Length);
                // Copy the contents of the sourceFile to the destFile.
                int bytesRead;
                // read 1K at a time
                byte[] buffer = new byte[1024];
                do
                {
                    // Read from the wrapping CryptoStream.
                    bytesRead = inStream.Read(buffer, 0, 1024);
                    outStream.Write(buffer, 0, bytesRead);
                } while (bytesRead > 0);
            }
        }
    }
    return;
} // end SignFile

// Compares the key in the source file with a new key created for the data portion of the file. If the
keys
// compare the data has not been tampered with.
public static bool VerifyFile(byte[] key, String sourceFile)
{
    bool err = false;
    // Initialize the keyed hash object.
    using (HMACSHA256 hmac = new HMACSHA256(key))
    {
        // Create an array to hold the keyed hash value read from the file.
```

```
        byte[] storedHash = new byte[hmac.HashSize / 8];
        // Create a FileStream for the source file.
        using (FileStream inStream = new FileStream(sourceFile, FileMode.Open))
        {
            // Read in the storedHash.
            inStream.Read(storedHash, 0, storedHash.Length);
            // Compute the hash of the remaining contents of the file.
            // The stream is properly positioned at the beginning of the content,
            // immediately after the stored hash value.
            byte[] computedHash = hmac.ComputeHash(inStream);
            // compare the computed hash with the stored value

            for (int i = 0; i < storedHash.Length; i++)
            {
                if (computedHash[i] != storedHash[i])
                {
                    err = true;
                }
            }
        }
    }
    if (err)
    {
        Console.WriteLine("Hash values differ! Signed file has been tampered with!");
        return false;
    }
    else
    {
        Console.WriteLine("Hash values agree -- no tampering occurred.");
        return true;
    }
  } //end VerifyFile
} //end class
```

**LICENÇA DE UTILIZAÇÃO DO ALGORITMO HMAC**

A definição e análise da construção do HMAC foi publicada pela primeira vez em 1996 em um artigo de Mihir Bellare, Ran Canetti e Hugo Krawczyk [BCK, 1996]. Neste trabalho os autores apresentaram uma inovação simples e prática de esquemas de autenticação de mensagens baseados em uma função hash criptográfica. O esquema HMAC, é comprovadamente seguro, desde que a

função de hash subjacente tenha algumas forças criptográficas razoáveis. Neste artigo os autores também mostram, de forma quantitativa, que os esquemas retêm quase toda a segurança da função hash subjacente. Ou seja, se a função hash é sabidamente forte contra os ataques criprográficos, então a função HMAC que a utiliza também o será. Isso quer dizer que o desempenho do HMAC é essencialmente o mesmo da função hash subjacente. Além disso, os autores usam a função hash como dispositivo, para que os códigos ou bibliotecas de código aberto amplamente disponíveis em diversos sistemas, possam ser usados para implementar o algoritmo HMAC de maneira simples e com baixo custo em qualquer implementação.

Posteriormente o algoritmo HMAC foi proposto pela IBM ao IETF - Internet Engineering Task Force em fevereiro de 1997 [RFC-2104, 1997] com o intuito de se tornar um padrão para a Internet. O documento inicial foi atualizado pelo NIST - National Institute of Standards and Technology dos EUA em março de 2011 [RFC-6151, 2011]. A partir daí surgiram diversas implementações do algoritmo HMAC para vários tipos de aplicações em plataformas abertas ou "Opensource", de forma que as implementações podem ser usadas e modificadas sem a necessidade do pagamento de direitos autorais ou licenças de software. Na Internet existem dezenas de exemplos de código aberto implementando funções HMAC em diversas linguagens de programação.

As políticas do IETF sobre Direitos de Propriedade Intelectual (DPI), como direitos de patente, em relação às tecnologias desenvolvidas no IETF são projetadas para garantir que os grupos de trabalho e participantes do IETF tenham o máximo de informações possível sobre quaisquer restrições de DPI em uma proposta técnica. As políticas também visam beneficiar a comunidade da Internet e o público em geral, respeitando os direitos legítimos dos detentores de DPI.

Em resumo: o uso de algoritmos e funções HMAC são simples e não dependem de licenças ou pagamento de direitos.

## APLICAÇÕES QUE UTILIZAM HMAC

Conforme já discutido o HMAC é um tipo de código de autenticação de mensagem (MAC – *Message Authentication Code*) amplamente utilizado na Internet e que é obtido executando uma função hash criptográfica combinada com uma chave compartilhada secreta sobre os dados a serem autenticados. Como qualquer código de autenticação de mensagem, ele é usado tanto para integridade de dados quanto para autenticação.

Há inúmeros exemplos de aplicações, de código aberto e proprietárias, que utilizam HMAC para garantir integridade e autenticidade. Por exemplo, quando se recebe um e-mail com um link de

redefinição de senha que é válido apenas por um determinado período e só pode ser usado uma vez, é possível com a utilização de uma função HMAC.

A verificação da integridade dos dados é fundamental para criar garantias entre as partes envolvidas em processos de comunicação na Internet. Alguns dos protocolos mais usados na Internet para transferência de dados, especificamente o **HTTPS, o SFTP e o FTPS**, e outros protocolos de transferência de dados, usam HMAC. Ou seja, toda vez que se utiliza o protocolo de acesso web seguro HTTPS, além do certificado digital, estamos utilizando, um protocolo HMAC para negociação de uma senha única temporária entre o navegador e o servidor. Assinaturas digitais são quase semelhantes aos HMACs, ou seja, ambos empregam uma função de hash e uma chave compartilhada. A diferença está nas chaves, ou seja, os HMACs usam chave simétrica (chave única), enquanto as assinaturas e certificados usam chaves assimétricas (duas chaves diferentes).

Outra aplicação interessante é a autenticação de dados enviados por aplicativos externos – normalmente qualquer cenário em que se forneça um serviço que tenha a figura de uma "**chave de API**" como explicado em [Martin, 2020]. Nesse caso, compartilha-se uma chave secreta comum com o usuário do aplicativo. O benefício adicional dessa abordagem é que os HMACs são computacionalmente baratos e não requerem muita memória, portanto, o que é muito adequado para este cenário, notadamente em APIs de "Internet das Coisas" (IoT) devido ao menor custo e maior velocidade se comparados aos certificados digitais.

Uma outra aplicação importante eque usa HMAC é o ***Google Authenticator*** conforme descito em [Chan, 2021]. Quando se habilita a autenticação de dois fatores em sites, eles geralmente mostram um código QR e solicitam que se digitalize este código e inicie seu aplicativo autenticador. Alguns sites pedem especificamente que se utilize o ***Google Authenticator***. O que o *Google Authenticator* usa são os algoritmos de senha de uso único com base em HMAC (*HOTP – HMAC One Time Password*) e senha de uso único com base em tempo (*TOTP – Time One Time Password*). Outros aplicativos autenticadores como ***Authy***, ***Duo Mobile***, ***Lastpass*** e ***1Password*** implementam os mesmos algoritmos e são capazes de gerar exatamente os mesmos tokens que você obtém do *Google Authenticator* usando HMAC.

Estas são apenas algumas dos exemplos de aplicações que utilizam HMAC. Há inúmeros outros exemplos em aplicações proprietárias, em sistemas financeiros, de logística, militares, dentre outros.

**CONCLUSÃO**

A utilização de certificados digitais para a liberação de relatórios de exames em laboratórios, conforme fundamentado nas sessões anteriores é, senão uma impossibilidade, uma quebra paradigmática no método hoje utilizado pelos laboratórios, em especial para atender ao princípio da escalabilidade.

Outrossim, a utilização de certificados digitais, hoje possível apenas para pessoas físicas, não se adequa ao processo pós-analítico neste documento descrito, em razão de a liberação dos diversos exames que podem ser solicitados, não acontecer concomitantemente, tampouco sua realização ser de responsabilidade de uma única pessoa.

Outra questão preponderante é que a utilização de certificados digitais, além de comprometer as fases pré-estabelecidas dos processos já existentes, o relatório produzido pelos laboratórios não se amolda ao conceito de laudo, o que torna inexequível a utilização de certificados digitais. Embora, a RDC 302/2005 tenha, no tópico 6.3.3, elencado todos os requisitos que devem compor um laudo laboratorial, não o faz do ponto de vista conceitual, vez que o conceito de laudo em muito se distancia do documento materializado na entrega de exames laboratoriais. Os laboratórios destinam aos pacientes, apenas e tão somente o resultado dos exames solicitados por algum médico e, em larga medida, o descrevem na forma de um relatório de resultados que são utilizados pelo solicitante na composição do laudo médico, vez que subjaz ao conceito de laudo a interpretação dos resultados dos exames para a composição do diagnóstico do paciente.

Claro está, portanto, que além da impossibilidade prática, considerado o processo de que dispõe os laboratórios para a realização dos exames e o conceito de que o documento disponibilizado aos pacientes se consubstancia em um relatório, não entendemos ser viável, tampouco fundamental a utilização de certificados digitais nos relatórios de exames entregues por laboratórios de análises clínicas.

Isto posto, salientamos ainda a possibilidade disposta na Lei 14.063/2020 sobre a utilização de assinaturas avançadas em documentos de saúde, desde que atendidos os requisitos de integridade e autenticidade da informação – art. 14, I, II.

Diante de todos os argumentos jurídicos e técnicos aqui apresentados, nosso parecer é pela aplicação de um algoritmo de *HMAC* de modo a permitir **verificar, simultaneamente, a integridade dos dados e a autenticidade de uma mensagem ou dados constantes em um exame**. O algoritmo de HMAC pode ser aplicado nos dados pessoais do cliente, e em cada resultado individual de teste

executado, antes desse ser encaminhado para a fase entrega de resultado. Fica a critério dos LIS padronizar quais campos deverão ser utilizados.

Este procedimento irá garantir 2 aspectos importantes: o primeiro, por intermédio da função de hash, irá **garantir INTEGRIDADE de que o resultado é fiel aos dados emitidos pelo laboratório** ou equipamento sem interferência humana que pudesse alterar o resultado automático, e o segundo, **garantir a AUTENTICIDADE de que aquele resultado foi realmente emitido pelo laboratório**, uma vez que só o laboratório possui a chave secreta a ser aplicada na função HMAC.

É nossa recomendação que deverá ser adotada uma padronização para a confirmação da integridade e autenticidade de uma determinada divulgação de resultado de exame, **por intermédio do algoritmo ou função HMAC-SHA256 disponível em praticamente todas as linguagens de programação atuais.** Esta funcionalidade deve ser capaz de apresentar o conjunto de dados que se encontra associado a um determinado hash, ou seja, ao conjunto de dados que foi usado para gerar o hash do exame em questão. Assim, caso necessário, o laboratório pode confrontar os dados que lhe são apresentados àqueles disponíveis no LIS e, dessa forma, obter as devidas conclusões e encaminhamentos. Fica a critério de cada LIS o mecanismo de gestão de chaves secretas utilizadas no algoritmo de HMAC.

E, ainda que enxerguemos esta dificuldade e a pouca valia de tal instrumento, propomos alternativamente, a aplicação de um algoritmo de HMAC a ser definido por parte das áreas técnicas de cada integrante da LisBrasil. Assim, o HMAC pode ser usado para verificar simultaneamente a integridade dos dados e a autenticidade de uma mensagem. O algoritmo de HMAC pode ser aplicado nos dados pessoais do cliente, e em cada resultado individual de teste executado, antes desse ser encaminhado para a fase de entrega de resultado. Este procedimento irá garantir dois aspectos importantes: o primeiro, por intermédio da função hash, fica garantida a integridade de que o resultado é fiel aos dados emitidos pelo equipamento sem interferência humana que pudesse alterar o resultado automático, e o segundo, garantir a autenticidade de que aquele resultado foi realmente emitido pelo laboratório, uma vez que só o laboratório possui a chave secreta a ser aplicada na função HMAC.

São Paulo, 14 de abril de 2022.

OPICE BLUM, BRUNO E VAINZOF ADVOGADOS

**REFERÊNCIAS BIBLIOGRÁFICAS - 1**

AGÊNCIA NACIONAL DE VIGILÂNCIA SANITÁRIA. Resolução da diretoria colegiada- RDC nº 302, de 13 de outubro de 2005. Disponível em: https://www20.anvisa.gov.br/segurancadopaciente/index.php/legislacao/item/rdc-302-de-13-de-outubro-de-2005 . Acesso em: 26, janeiro 2021.

AGGARWAL S, GOYAL N, AGGARWAL K. *A review of comparative study of MD5 and SHA security algorithm*. Int J Comput Appl. 2014;104:1–4.

BRASIL, Governo Federal. Portal Da Legislação. Código de Defesa do Consumidor. Disponível em: http://www.planalto.gov.br/ccivil_03/leis/l8078compilado.htm . Acesso em: 21, janeiro de 2021.

*Deciphering Your Lab Report*. LAB TESTS ONLINE, 2020. Disponível em: https://labtestsonline.org/articles/how-to-read-your-laboratory-report . Acesso em: 26, janeiro 2021.

*Lab Oversight: A Building Block of Trust*. LAB TETS ONLINE,2019. Disponível em: https://labtestsonline.org/articles/laboratory-regulation . Acesso em: 26, janeiro 2021.

M. Bellare, R. Canetti, and H. Krawczyk. *Keyed Hash Functions and Message Authentication*. Proceedings of Crypto'96, LNCS 1109, pp. 1-15. (http://www.research.ibm.com/security/keyed-md5.html).

Quando automatizar o laboratório de análises clínicas?. AUTOLAC. Disponível em: https://autolac.com.br/blog/quando-automatizar-o-laboratorio-de-analises-clinicas/ . Acesso em: 26, janeiro 2021.

RFC-2104 - H. Krawczyk, M. Bellare e R. Canetti. HMAC: *Keyed-Hashing for Message Authentication*. https://datatracker.ietf.org/doc/html/rfc2104

UNITED STATES. 42 U.S.C, §493. *Eletronic Code Of Federal Regulations*, 1988. Disponível em: https://ecfr.federalregister.gov/current/title-42/chapter-IV/subchapter-G/part-493 . Acesso em: 26, janeiro 2021.

**REFERÊNCIAS BIBLIOGRÁFICAS – 2**


| | |
|---|---|
| [Aggarwal, 2014] | Aggarwal S, Goyal N, Aggarwal K. *A review of comparative study of MD5 and SHA security algorithm*. *Int J Comput Appl.* 2014;104:1–4. |
| [BCK, 1996] | Mihir Bellare, Ran Canetti & Hugo Krawczyk. *Keying Hash Functions for Message Authentication*, CRYPTO 1996, pp. 1–15. https://cseweb.ucsd.edu/~mihir/papers/kmd5.pdf (Verificado em 07/01/2022) |
| [Bellare, 1996] | M. Bellare, R. Canetti, and H. Krawczyk. "Keyed Hash Functions and Message Authentication", Proceedings of Crypto'96, LNCS 1109, pp. 1-15. http://www.research.ibm.com/security/keyed-md5.html. (Verificado em 07/01/2022). |
| [Chan, 2021] | Jeremy Chan. How Google Authenticator, HMAC-Based One-time Password, and Time-based One-time Password Work. Level Up Coding. 28 february 2021. https://levelup.gitconnected.com/how-google-authenticator-hmac-based-one-time-password-and-time-based-one-time-password-work-17c6bdef0deb (07/01/2022) |
| [hashlib, 2022] | Python V.3.10.1, *Módulo hashlib — Secure hashes and message digests*. Disponível em https://docs.python.org/3/library/hashlib.html (07/01/2022) |
| [hmac, 2022] | Python V.3.10.1 hmac — Keyed-Hashing for Message Authentication. Disponível em https://docs.python.org/3/library/hmac.html#module-hmac (Verificado em 07/01/2022) |
| [Martin, 2020] | Jarred Martin. *Verifying App-API communication using an HMAC*. 21 Apr. 2020. Medium. https://medium.com/@jarredmartin/secure-app-api-communication-using-an-hmac-316314b63445 (07/01/2022) |
| [Menezes, 1996] | Menezes, Alfred J.; van Oorschot, Paul C.; Vanstone, Scott A (1996). *Handbook of Applied Cryptography*. CRC Press. ISBN 978-0849385230. |
| [Paar, 2014] | Christof Paar, C. & Pelzl, J. *Understanding Cryptography, A Textbook for Students and Practitioners*. Springer. 10a. Ed. (2014). ISBN: 3642446493 |

| | |
|---|---|
| [RFC-1321, 1992] | Rivest, R. The MD5 Message-Digest Algorithm. Request for Comments: 1321. MIT Laboratory for Computer Science and RSA Data Security, Inc. (1992) https://www.ietf.org/rfc/rfc1321.txt (07/01/2022) |
| [RFC-2104, 1997] | RFC-2104 - H. Krawczyk, M. Bellare e R. Canetti. *HMAC: Keyed-Hashing for Message Authentication*. (1997) https://datatracker.ietf.org/doc/html/rfc2104 (Verificado em 07/01/2022) |
| [Schneier, 1995] | Schneier, B. *Applied Cryptography: Protocols, Algorithms, and Source Code in C*. Wiley; 2nd ed. edição (1995), ISBN 978047111709: |
| [Wiki, 2021] | Wikipedia, the free encyclopedia. *Message Authentication Code*. 2021. https://en.wikipedia.org/wiki/Message_authentication_code#/media/File:MAC.svg (07/01/2022) |
| [Wiki, 2022] | Wikipedia, the free encyclopedia. 2022. *HMAC*. https://en.wikipedia.org/wiki/HMAC (07/01/2022) |