Internet Draft
Document: draft-hallambaker-pcon-00.txt
Expires: December 2006

P. Hallam-Baker
VeriSign Inc.

June 2006

Considerations for DKIM Policy Language

Status of this Memo

Abstract

If we are going to change the current DKIM policy language as deployed in any way including definition of a new DNS RR we should do the job properly and define a layered infrastructure that is capable of other uses.

Contents

1. Conventions used in this document

The key words "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT", "SHOULD", "SHOULD NOT", "RECOMMENDED", "MAY", and "OPTIONAL" in this document are to be interpreted as described in RFC-2119 [RRC 2119].

2. Is change necessary?

Sender Security Policy (SSP) appears to be an adequate basis for a security policy to support the immediate needs of DKIM. It appears that the email sender has sufficient scope to describe the configuration of their outbound email authentication using the DKIM message format and it is is clearly capable of extension to support additional attributes that may be found to be necessary.

An entirely reasonable approach to supporting the policy needs of DKIM is to simply adopt the SSP framework as is maintaining strict backwards compatibility but making whatever minor adjustments as may be found necessary.

The adoption of a new DNS Resource Record is not supported in any existing DKIM infrastructure and thus represents an incompatible change.

If however backwards compatibility is to be lost for any reason the working group should reject the current SSP document which is exclusively focused on DKIM and instead adopt a layered approach in which the working group first defines a framework for expressing general security policy and then defines the DKIM policy mechanism as an instance of that framework.

DKIM is an important resource for managing email security. The lack of robust, practical, ubiquitous email authentication is clearly the most significant defect in the present email system. It is not however the only security defect in the email system that needs to be fixed. For many years large numbers of email servers have been capable of supporting email exchange over TLS, without policy layer support however this infrastructure is vulnerable to a downgrade attack. The use of domain level S/MIME and PGP face the same problem.

The need for a policy layer integrated into the signaling system represents a clear deficiency in the Internet infrastructure and is one of the principal causes for many of the limitations that are felt when using or administering Internet protocols. Most protocols provide support for version numbers but as is widely acknowledged a version number does little more than avoid unintended consequences when a client attempts to connect to a server using an incompatible protocol version. Administration of a transition from one protocol version to another or from one protocol to a different one requires a policy infrastructure.

The Internet is gradually acquiring a policy infrastructure and a decision by the DKIM working group to decide to adopt or reject a layered approach will have little impact on this process. A policy infrastructure is already being developed to support Web Services, the need to extend this framework to support REST or AJAX based applications is inevitable.

Adopting a layered approach in DKIM will not change the fact of policy infrastructure deployment but it may impact the consistency and comprehensiveness of the infrastructure that evolves. A similar effect was seen in the deployment of MX and SRV. The need to provide fault tolerant service was felt first in email. Later it was realized that this need was inherent in every Internet protocol and the SRV record was defined.

3. What is a Layered Infrastructure?

The principle of layered abstraction is a basic tool of network design yet providing a definition of a layered abstraction is almost challenging as designing one.

The statements we want to make for DKIM are instances of more general requirements:

> "Every email from example.com has a DKIM signature with characteristics {x, y, z}"
> "The example.com email server always offers STARTTLS with charateristics {p, q}"
> "http://example.com supports HTTP/2.0"

Instead of defining statements and syntax that are specific to DKIM we should attempt to define policy statements in such a way that encourages reuse whenever possible.

3.1 What a Layered Architecture is Not

Equally important is the definition of what a layered architecture is not. The key to the design of a layered architecture is the specification of a series of well defined abstractions and the definition of the communication between those abstractions.

If the definition of the policy language requires constant recourse to make changes to the DNS protocol the policy language framework is broken.

The DNS is a large deployed infrastructure with a very specific task. Changes to the DNS infrastructure to support deployment of a policy infrastructure should be avoided if at all possible but are acceptable for the purposes of deploying a policy infrastructure that will serve multiple protocols.

What is not acceptable is a ‘policy infrastructure’ where each policy definition to support a new protocol requires changes to be made to the DNS infrastructure.

3.2 Reusable Escape Mechanism

Clearly there is a strict limit to the detail that is practical within the context of DNS publication of service configuration data, clearly we do not want people entering war and peace into the DNS to configure an email service. It is bad enough the fact that airline check in assistants have to write novels while checking people in for a flight from Boston to Washington without network administrators attempting to write complex security policies into DNS configuration files to be emitted as 500 byte DNS response messages.

Writing security policy should not be like writing a haiku: fitting a complex idea into a tightly restricted form of expression.

There are two means of extension that might be employed. First the policy mechanism might extend within the DNS itself using some form of include directive similar to that defined in SenderID/SPF framework. Alternatively the extension might be by means of an arbitrary URL from which the full policy may be obtained.

If the extension mechanism is to be to an arbitrary URL it is probably most appropriate to allow use of a richer, more expressive syntax such as XML than the compact tag value encoding forms generally considered more appropriate for use in the DNS.

3.3 Reusable Cryptography Attributes

A security policy is at root a statement that says that a certain network principle shall always offer a certain minimum level of security when making a communication.

For example

> ‘example.com will always sign outgoing emails using a minimum of SHA-1 and RSA1024’

‘example.com will always offer STARTTLS in SMTP transactions with a certificate validated via a cert chain whose root cert has a SHA1 fingerprint of Q282hd9213h23ey23== and offer a minimum of RSA 1024 and RC4’

It is clear that many of the attributes referenced in the above policies will be generally applicable. For example IANA maintains a registry of consistent names for cryptographic algorithms.

Note that the above policy statements need not exhaustively enumerate every cryptographic algorithm that might be offered. If the email server in the second example offers AES 128 it need not mention that it also supports 3DES and AES 256.

It is however essential that each of the algorithms described be offered in the form described. Otherwise there would be the possibility of a downgrade through unacceptable upgrade attack where a man in the middle inserts an offer of RSA 32768 knowing that it will be rejected even though it is stronger than RSA 1024.

3.4 Reusable Discovery Strategy

The principle area in which the current SSP specification is unsatisfactory is in the area of policy discovery. It appears that unless a new RR is defined specifically for DKIM policy records that it will be impossible to support the form of wildcarding we require without resort to heuristics or exhaustive search strategies that may facilitate denial of service attacks.

Definition of a new RR should be avoided if at all possible. According to source code samples demonstrated by one of the leading providers of the deployed base of DNS servers their product is not capable of saving data associated with unknown RRs. Any configuration changes made to the server by mechanisms such as the dynamic DNS component which does support use of new RRs will thus be lost whenever the service is stopped and restarted.

In the absence of proof that the DNS server vendor concerned engaged in a deliberate lie it is prudent to assume that any deployment of new DNS RRs has a significant infrastructure cost and should be avoided unless absolutely necessary.

As we know the DNS wildcard scheme is not ideal for our purposes:

1: There is no way to specify a wildcard of the form _prefix.*.example.com

2: The prefix *.example.com does not match host1.example.com if there is any record defined for that node

The solution recommended by the DNSEXT Working Group members to the first problem is to define a new DNS RR.

The second problem exists whether or not a new record is defined. The only way to address this problem within the existing DNSSEC framework is to add support at the DNS server level for synthetic

wildcards. So the admin enters a policy for ?.example.com which is expanded out to create records for every host in example.com that does exist and does not have a policy record otherwise defined.

The first problem is the hard one, one solution is to define a new resource record. This is not sustainable as an infrastructure. Every internet protocol will need a DNS policy record associated with it so we would need to define 10,000 new RRs just to support existing protocols.

A better solution is to define a record to act as the wildcard record. The use of the PTR record is currently undefined for the forward DNS and allows the needed information to be defined. Alternatively a new record could be specified, but this would then make implementations dependent on the deployment of DNS extensions.

The policy discovery strategy then becomes:

To discover the policy for DKIM at alice.example.com:

```
1) policy = lookup (TXT, "_dkim.alice.example.com")
        IF policy <> NULL THEN RETURN policy

2) pointer = lookup (PTR, "alice.example.com")
        IF pointer == NULL THEN RETURN NULL

3) policy = lookup (TXT, "_dkim." + pointer)
        return policy
```

This strategy is guaranteed to always return in three steps without exception and is 100% compatible with the deployed DNS infrastructure with no known exceptions.

The strategy is can be applied to an arbitrary protocol and allows for much simplified administration. In most networks the majority of the host configurations are essentially identical. Only a few machines that perform special roles such as mail handling or file server support will require custom configuration.

The redirect strategy allows the administrator to define standard network policy configurations, for example in an enterprise there would probably be defined network policy configs for standard desktops, standard laptops and developer machines. If necessary the standard config may be overriden for individual domain names.

The redirect stategy does not depend on walking up the chain, finding a zone cut or anything similar, it also overcomes a frequent objection to such attempts, it is does not allow an unauthorized intermediate DNS server to override policy definitions made by a subordinate zone - except of course to the extent that it can do this by changing the delegation and hijacking the entire zone.

A possible objection to the strategy is that it imposes new semantics on the PTR record, a record which to date has only been defined for use in the reverse DNS. Fortunately however the lack of defined semantics outside the reverse DNS means that the only problem that may occur in the forward DNS is that this use may collide with other attempts to redefine PTR.

The issue of the reverse DNS is more complex. The most prudent course might be to prohibit the use of PTR for policy redirection, yet the semantics that are achieved through use of PTR appear to be exactly what we might want such semantics to be. It is probably prudent to simply note this apparent oddity and note that since email is not sent from the reverse DNS zone it is unlikely to impact DKIM.

4. Next Steps

4.1 Attempt a Layered Definition of SSP

We recommend that the DKIKM working group examine the possibility of redefining the current SSP specification in terms of a layered model where the policy infrastructure framework is separated from the instance to support DKIM.

4.2 New RR Definitions Should Support Infrastructure, not DKIM

A previously noted a policy infrastructure should separate the specific needs of DKIM from needs that are general to multiple protocols.

If it is the case that a new RR definition is essential and it is not possible to deploy DKIM in a manner that meets all the objectives of DKIM within the capabilities of the legacy DNS then whatever RRs are defined should be designed to support a policy infrastructure rather than the specific DKIM protocol.

It is simply not acceptable or sustainable for all attempts to define protocol policy records to be gated on a single IETF working group, particularly one that will in any case be wound up as soon as the DNS Security deployment is complete. There are many forums that define Internet protocols and every protocol will require policy.

There are already far more unofficial definitions of DNS SRV entries than there are official ones. Unless a realistic approach is adopted that supports the principal of independent innovation once considered the founding principle of the IETF as an institution.

Notices

References

[RFC 2119] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, March 1997

Acknowledgments

The author would like to acknowledge the contribution of Stephen Farrell without whose insistence on writing it this document would never have been written.

Authors' Addresses

Phillip Hallam-Baker
VeriSign Inc.
pbaker@verisign.com