DOMAIN NAME SYSTEM

SECURITY TECHNICAL IMPLEMENTATION GUIDE

Version 2, Release 2

11 March 2005

Developed by DISA for the DOD

This page is intentionally left blank.

TABLE OF CONTENTS

Page

This page is intentionally left blank.

SUMMARY OF CHANGES

The Domain Name System (DNS) STIG includes numerous changes relative to the last version of the Section 1. These changes were made to Version 1, Release 1, dated 19 March 2004.

GENERAL

Updated cover page, headers, and footers with the newest version of the Section 1 template.

SECTION 1

Made changes based on STIG consistency efforts.

1. INTRODUCTION

The Domain Name System (DNS) is a hierarchically structured, distributed database that provides name resolution services to Internet Protocol (IP) networks such as the Internet. The most common use of DNS is to map a host name to its network address. For example, the host name www.cert.mil is mapped via DNS to 199.211.123.12, the dotted decimal notation for its 32-bit IP address.

DNS also supports reverse lookup, in which a network address is mapped to a host name. Other supported directory capabilities include lookup of name servers and mail exchangers. Emerging technologies such as IPv6 and wireless Internet also depend on DNS for basic functionality.

The security posture of DNS within the Department of Defense (DOD) is of concern for several reasons. The data it provides must be highly reliable and available to support both peacetime and wartime requirements; and the dissemination of DNS information must not itself introduce security risks into the DOD computing environment. Specifically, mechanisms must exist for name servers to verify the accuracy and origin of DNS information received.

Unfortunately, DNS has many security weaknesses. Early implementations of DNS did not provide for transmission authentication or guarantee the integrity or accuracy of the global database, which left the DNS service open to attacks. More recent versions of DNS software provide greater information assurance (IA), but it must be appropriately configured to achieve this assurance. Without reliable DNS, nearly all DOD information systems are left vulnerable, including mission critical applications such as the Global Command Control System (GCCS) and the Defense Messaging System (DMS).

This *DNS Security Technical Implementation Guide* (STIG) is designed to assist administrators with the configuration of DNS server software and related portions of the underlying operating system. This STIG also provides guidance for standard operating procedures related to configuration management, business continuity, and other topics.

1.1 Background

This document supersedes previous DNS guidance found in the Network and UNIX STIGs. It also has implications for organizations within the scope of the Enclave, IP WAN, and Web Server STIGs.

It should be noted that FSO support and Technical Support for the STIGs, Checklists, and Tools are only available to DOD Customers.

1.2 Authority

DOD Directive 8500.1 requires that “all IA and IA-enabled IT products incorporated into DOD information systems shall be configured in accordance with DOD-approved security configuration guidelines” and tasks DISA to “develop and provide security configuration guidance for IA and IA-enabled IT products in coordination with Director, NSA.” This document is provided under the authority of DOD Directive 8500.1.

The use of the principles and guidelines in this STIG will provide an environment that meets or exceeds the security requirements of DOD systems operating at the MAC II Sensitive level, containing sensitive information.

1.3 Scope

This document details DOD DNS security practices and procedures applicable to all DOD Top Level Domain (TLD) and below name servers. The requirements set forth in this document are designed to assist Information Assurance Officers (IAOs) and DNS administrators. IAOs were formerly called Information Systems Security Officers (ISSOs). The policy portions of this STIG are relevant to all name servers connected to either the DOD Non-Classified Internet Protocol Router Network (NIPRNet) or Secret Internet Protocol Router Network (SIPRNet).

This document supports the following implementations of DNS:

- BIND 9.2.1
- BIND 9.2.1 for Microsoft Windows NT, Windows 2000, and Windows 2003
- Windows 2000 DNS
- CSS DNS

The field use of Berkeley Internet Name Domain (BIND) releases are 8.2.7 and above, 8.3.4 and above, and 9.2.1 and above are acceptable, but the document does not detail the syntax of BIND 8 configuration statements. In most cases, BIND 8 and BIND 9 statements are identical. When they are not, organizations deploying BIND 8 must make the appropriate changes to the BIND 9 syntax in this document to achieve the desired effect.

The field use of Windows 2003 DNS is also acceptable, but the document does not include the configuration of Windows 2003 dialog boxes. Nevertheless, security relevant configuration options for Windows 2000 and Windows 2003 DNS are nearly identical. Any references to Windows 2000 in this document also apply to Windows 2003.

This STIG does not address the DNS configuration of DNS clients (i.e., the workstations, servers, and network devices that query name servers). Each of these clients runs DNS resolver software. Any requirements concerning those resolvers would be addressed in the STIG corresponding to the underlying technology (e.g., Desktop, Network, etc.).

Adherence to the standards delineated in this document assures compliance with the *DISA Action Plan for DOD DNS Hardening*.

1.4 Writing Conventions

Throughout this document, statements are written using words such as "**will**" and "**should**." The following paragraphs are intended to clarify how these STIG statements are to be interpreted.

A reference that uses "**will**," indicate mandatory compliance. All requirements of this kind will also be documented in the italicized policy statements in bullet format, which follow the topic

paragraph. This makes all "**will**" statements easier to locate and interpret from the context of the topic. The IAO will adhere to the instruction as written. Only an extension issued by the Designated Approving Authority (DAA) will table this requirement. The extension will normally have an expiration date, and does not relieve the IAO from continuing their efforts to satisfy the requirement.

A reference to "**should**" indicates a recommendation that further enhances the security posture of the site. These recommended actions will be documented in the text paragraphs but not in the italicized policy bullets. Nevertheless, all reasonable attempts to meet this criterion will be made.

For each italicized policy bullet, the text will be preceded by parentheses containing the italicized Short Description Identifier (SDID), which corresponds to an item on the checklist and the severity code of the bulleted item. An example of this will be as follows "(*G111: CAT II*). "If the item presently has no Potential Discrepancy Item (PDI), or the PDI is being developed, it will contain a preliminary severity code and "N/A" for the SDID (i.e., "*[N/A: CAT III]*").

1.5 DISA Information Assurance Vulnerability Management (IAVM)

The DOD has mandated that all IAVMs are received and acted on by all commands, agencies, and organizations within the DOD. The IAVM process provides notification of these vulnerability alerts and requires that each of these organizations take appropriate actions in accordance with the issued alert. IAVM notifications can be accessed at the Joint Task Force - Global Network Operations (JTF-GNO) web site, http://www.cert.mil.

1.6 Vulnerability Severity Code Definitions

Category I	Vulnerabilities that allow an attacker immediate access into a machine, allow superuser access, or bypass a firewall.
Category II	Vulnerabilities that provide information that have a high potential of giving access to an intruder.
Category III	Vulnerabilities that provide information that potentially could lead to compromise.
Category IV	Vulnerabilities, when resolved, will prevent the possibility of degraded security.

Table 1-1. Vulnerability Severity Code Definitions

1.7 STIG Distribution

Parties within the DOD and Federal Government's computing environments can obtain the applicable STIG from the Information Assurance Support Environment (IASE) web site. This site contains the latest copies of any STIG, as well as checklists, scripts, and other related security information. The NIPRNet URL for the IASE site is http://iase.disa.mil/.

1.8 Document Revisions

Comments or proposed revisions to this document should be sent via e-mail to fso_spt@disa.mil. DISA FSO will coordinate all change requests with the relevant DOD organizations before inclusion in this document.

This page is intentionally left blank.

2. DNS TERMINOLOGY AND CONCEPTS

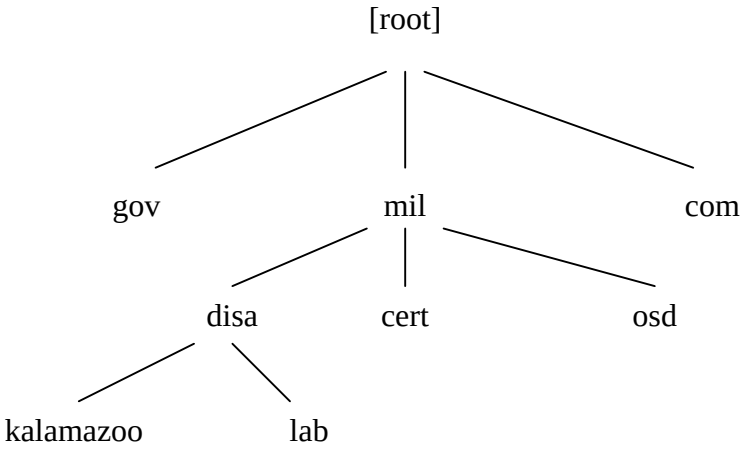
DNS is a hierarchically structured, distributed database that provides name services to IP networks such as the Internet. This database contains hostnames and their corresponding IP addresses. Name servers support hostname to IP address translation (forward resolution) as well as the reverse, IP address to hostname translation (inverse resolution). The DNS protocol establishes rules for the following:

- Querying and updating the names database
- Database replication
- Database schema

DNS has evolved into a critical component of the Internet infrastructure by making network navigation less cumbersome for the user and applications on large IP networks. The following sections describe major DNS concepts, components, and security objectives.

2.1 Domains

The data stored in the DNS is identified by *domain names* that are organized as a hierarchical tree according to organizational or administrative boundaries. A hypothetical domain tree might be as follows:



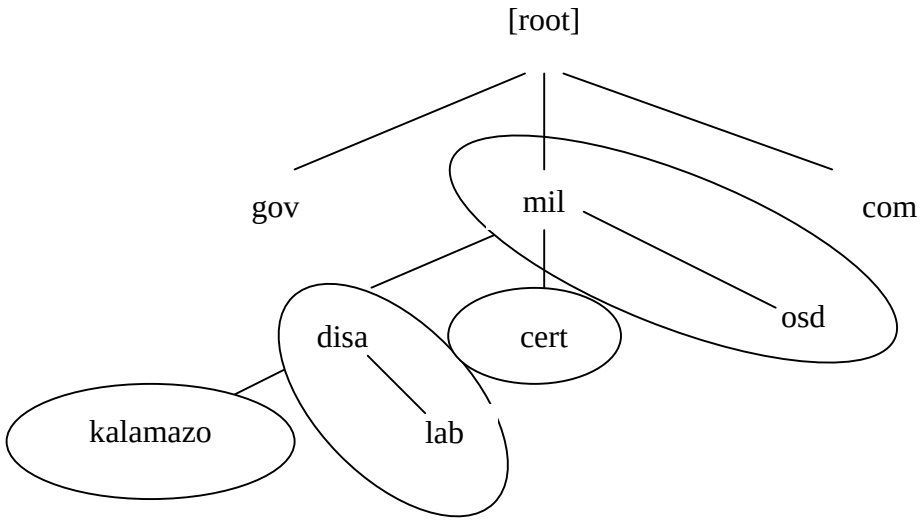
Each branch (or node) on the tree is given a *label*. A fully qualified domain name is the concatenation of all these labels on a given path from the domain to the root. Typically, domain names are written as a string of labels listed from right to left and separated by periods. For example, consider the following fully-qualified domain name:

kalamazoo.disa.mil.

In the example, kalamazoo is a third or lower level domain, disa is a second level domain, and mil is a Top Level Domain (TLD). Note that mil is followed by a period. This represents the root but is usually omitted in most applications today because all Internet domains share the same root.

2.2 Zones

Most users of IP-based applications are very familiar with domains and their use, but *zones* actually are the more relevant building blocks of the naming structure in DNS. A zone is a logical group of network devices and may be either an entire domain, a domain with all of its subdomains, or a portion of a domain. In our hypothetical example, the relationship of domains and zones might be as follows:



In this example, the kalamazoo and cert domains are associated with their own zones. The zones for disa and mil, on the other hand, are combined with the sub-domains lab and osd, respectively. There are many different ways to partition any domain tree into various zones.

The rationale for the existence of zones is that they make it easier to distribute the name database. Just as it would not be practical for the entire Internet DNS to reside on a single server, it also does not make sense to assign a unique name server for each lower-level domain. Zones allow network architects and administrators to combine domains in ways that optimize the management of a given portion of the name space.

2.3 Name Servers

A *name server*'s primary function is to respond to client queries for information from the DNS. Although name servers can be configured in a wide variety of ways, there are essentially two types of name servers:

- Those that store zone records for one or more DNS zones.
- Those that search for records stored on other name servers.

The first type of name server is called an *authoritative* name server because it can answer authoritatively when asked for information about records in one of its zone files. It consults its internal data to answer incoming requests rather than trying to pass them on to another server. In some references, authoritative name servers are also called *advertising* name servers.

The second type of name server is called a *caching* name server because it caches the queries it resolves on behalf of clients. Most computers on the Internet do not search for and query authoritative name servers directly, but instead are configured to query from one to three caching name servers that do this work on their behalf. Caching name servers are also referred to as *recursive* or *resolving* name servers. In particular, DISA Application Engineering (API) documents use the term recursive name server in this context.

Importantly, a name server can be simultaneously both an authoritative and a caching name server, but this is not considered a secure configuration. When an authoritative name server is also a caching server, both authoritative and external records are stored in the cache. This configuration means that if an adversary is able to corrupt the cache through external queries, the adversary may be able to modify the name server's responses to queries for authoritative records. When the authoritative function is separated from the resolving function – as good practice dictates – this risk is eliminated.

2.3.1 Authoritative Name Servers

For fault tolerance, most zones have two or more authoritative servers. The authoritative server, where the master copy of the zone data is maintained is called the *primary master* server, hereafter simply called the *master*. It loads the zone contents from a local zone file that the DNS database administrator creates and edits.

Each master for a zone may have one or more associated *secondary master* servers, hereafter called *slaves*. A slave server is also authoritative for that zone but gets its updates from its master server using a replication process known as a *zone transfer*. Typically, each zone has only one master from which all slaves get updates, but configurations exist in which one or more of the slaves themselves serve as masters to other slaves.

2.3.2 Caching Name Servers

A caching name server can resolve client queries using one of two mechanisms:

- Forwarding
- Recursion

In *forwarding*, the name server simply forwards the request on to another caching server without any additional effort. With *recursion*, the name server recursively searches for another name server that has authoritative name information to get the records that the client requested.

In this document, servers that support forwarding are referred to as *forwarding* servers. Similarly, servers that support recursion are called *recursive* servers.

Caching servers store the results of the lookups they perform in memory. This process, known as *caching*, improves performance because frequently queried records are readily available. Caching also reduces network traffic, since the zone's authoritative name servers do not need to be queried as often.

When a caching name server attempts to resolve a host name, it first searches its cache for the lowest level information in the DNS hierarchy it can find. For example, if trying to resolve kalamazoo.disa.mil, the name server first checks if there is an A record for that name in the cache. If there is not an A record, then the name server checks whether there is an NS record for kalamazoo.disa.mil in the cache. If there is an NS record, then it sends a query to the authoritative server for kalamazoo.disa.mil. If there is not an NS record, then it searches for an authoritative name server for disa.mil and so on.

Upon boot up, the caching name server has no previously stored information in its cache and thus must start with the root authoritative name servers. Knowledge of the root name servers is located in the root hints, a file that is included in the installation of the name server software and can be updated if IP addresses of the root servers change.

2.4 Name Server Support Roles

Supporting a name server involves administration of three separate components:

- The underlying operating system
- The name server software
- The DNS zone database

In this document, the term *System Administrator* or SA refers to the individual responsible for operating system functions, and the term *DNS software administrator* refers to the individual responsible for administering DNS software (e.g., modifying configuration files) and the term *DNS database administrator* refers to the individual responsible for entering, modifying, and deleting records in the DNS zone database. When the term *DNS administrator* appears, it refers to both DNS software and DNS database administrators.

At some sites, an individual will be responsible for all these functions, while at other sites responsibility for these functions will be divided among several individuals or teams. Each site determines the manner in which it separates or assigns duties, but regardless of the division of labor, all supporting personnel must comply with this STIG. To facilitate Security Readiness Reviews (SRRs), the IAO will be expected to explain how DNS support functions are delineated among personnel at that location in order that the reviewer can work with appropriate individuals to complete his or her work.

2.5 Resolvers

Another key component of DNS is the client *resolver,* which formulates and sends DNS queries to name servers. Most resolvers are rather simple. They merely send host names to caching name servers and wait for the response. The resolvers typically are configured with a list of two or three caching name servers, thereby ensuring there is a backup in case the master name server is unavailable.

In the future, resolvers may perform more complex functions. For example, they may sign queries if the caching name server requires authentication of requests. They may also request that replies be signed and the signatures be validated. This functionality is not available in common commercial resolvers today.

This page is intentionally left blank.

3. DNS ARCHITECTURE AND GENERAL SECURITY REQUIREMENTS

DNS security can be addressed at many different levels ranging from enterprise architecture to operating system parameters. Some aspects of the DNS protocols require strict consistency across all DNS products and operating systems – otherwise the wide variety of hosts on the Internet and within enterprise networks could not discover the host names and IP addresses they need to function properly. Other aspects of DNS, particularly with regard to security configuration, allow for differences among DNS software products.

This section addresses issues that impact all DNS implementations regardless of the product selected or the operating system on which it runs.[1] It also provides guidance on what products are acceptable in the DOD computing environment. Later sections in the STIG provide additional guidance for specific implementations of DNS (i.e., BIND, Windows 2000 DNS, and Cisco CSS DNS).

3.1 Name Server Software

At the present time, BIND 9 is the only DNS implementation that supports all of the current and planned required security functionality. DOD has a large investment in the development of BIND 9, for which DISA rigorously tests new releases. BIND is free and, therefore, does not impose licensing costs on organizations that deploy it. Given these advantages, organizations must have a more compelling reason for use of a DNS alternative than just past experience with that alternative. Importantly, BIND 9 is a complete rewrite of an earlier version of BIND and was developed with security in mind. Recently published BIND vulnerability notices have not impacted users of BIND 9.

Some situations exist in which a BIND alternative may be acceptable. For example, the use of Windows 2000 DNS greatly reduces the complexity of administering Windows 2000 networks. Guidelines for the use of this alternative are delineated in *Section 5, Configuration of Windows 2000 DNS*, of this document.

There are also requirements associated with particular high-availability server implementations that are not met with BIND 9. For example, DISA Computing Services (CS) requires that DNS automatically remove records for a member of a server cluster on a temporary basis when that machine is offline and then return the record when operations resume. When servers are distributed over a wide area, DISA would also like its DNS to direct a client computer to the server that provides the fastest response time for that client, which could vary depending upon geographic proximity, network traffic, or server loads. DISA has selected Cisco CSS DNS to offer this functionality, which is not present in BIND 9. Guidelines for the use of this alternative – which is currently limited to DISA csd.disa.mil domain – are delineated in *Section 6, Configuration of Cisco CSS DNS*, of this document.

[1] The exception to this rule is Cisco CSS DNS, which does not support all the requirements in this section but nonetheless has been approved for specialized applications.

If an organization believes that it requires name server software other than those discussed above, then it should contact DISA FSO to determine an appropriate course of action. In most cases, use of an alternative will generate a SRR finding, although extensions may be granted when either it meets the general requirements listed in this section or mitigating controls provide equivalent levels of assurance as BIND.

The requirement is as follows:

- *(DNS0400: CAT II) The IAO will ensure that name server software on production name servers is BIND or Windows 2000 (or later) DNS unless operational requirements cannot be met with these options, in which case the alternative must still be configured in a manner to satisfy the general security requirements listed in this STIG.*

The only alternative currently permitted is Cisco CSS DNS, which is limited to hosts defined in the csd.disa.mil domain. CSS DNS is both subject to the some of these general security requirements, where applicable, and the specific STIG guidance for this product.

3.2 Name Server Operating System Platforms

Another critical component of DNS security is the security of the operating system (OS) platforms on which the DNS software runs. If it is not possible to secure that OS, then DNS itself cannot be secure. Accordingly, organizations must select an appropriate OS for its name servers, one that has a well-documented secure configuration.

Even a securely configured operating system is vulnerable to the flaws of the programs that run on it. To prevent DNS software from being subject to the vulnerabilities of other programs, it is best not to run other programs at all – or at least run only those programs that are necessary for either OS or DNS support. In other words, in a properly secured environment, a name server would not run on the same device that also provides users web, e-mail, or database services.

The requirements are as follows:

- *(DNS0410: CAT II) The IAO will ensure that DNS server software runs only on approved operating systems.*

- *(DNS0415: CAT II) The IAO will ensure that DNS software runs only on dedicated hardware, with the exception of Windows 2000 DNS, which may run on domain controllers that also host Active Directory services.*

- *(DNS0420: CAT II) The SA will ensure that only the DNS software process ID (PID) has read access to the files containing DNS encryption keys.*

- *(DNS0425: CAT II) The SA will ensure that only the DNS software PID and/or the DNS database administrator have edit access to the zone database files.*

- *(DNS0430: CAT II) The SA will ensure that only the DNS software administrator and the DNS software PID have read access to the DNS software configuration files and that only the DNS software administrator has write access to these files.*

- *(DNS0435: CAT II) The SA will ensure that the name server's IP address is statically defined.*

- *(DNS0440: CAT II) The SA will ensure that an integrity checking tool is installed and monitoring for any modifications to the root.hints and named.conf files.*

For detailed configuration guidance for DNS encryption keys within BIND on UNIX, refer to *Appendix E.3.2, TSIG Keys.*

For a detailed list of permitted services on a UNIX name server running BIND, refer to *Appendix E.1, Network Services.*

For a detailed list of permitted services on a Windows name server running ISC BIND, refer to *Appendix F.1, Network Services.*

As of publication of this STIG, the approved operating systems for DNS name servers are:

- Solaris 2.5.1 and above
- HP-UX 10 and above
- Windows NT, Windows 2000, Windows 2003, and above

Cisco CSS runs on its own Internetworking Operating System (IOS) and therefore is not subject to many of these requirements. Updates to this list prior to the next publication of the STIG may appear in the *DNS Security Checklist*, a companion document to this STIG that provides instructions on how to perform a DNS Security Readiness Review (SRR).

The technical implementation guides associated with the approved operating systems are as follows:

- UNIX STIG
- NSA Windows 2000 Security Recommendation Guides
- Guide to Securing Microsoft Windows NT Networks
- Addendum to the NSA Guide to Securing Microsoft Windows NT Networks
- NSA Guides to Securing Windows 2000

3.3 Redundancy, Dispersal, and Availability

A critical component of securing an information system is ensuring its availability. The best way to ensure availability is to eliminate any single point of failure in the system itself and in the network that supports it.

Fortunately, the inherent design of DNS supports a high-availability environment. Master and slave servers regularly communicate zone information, so if any name server is disabled at any time, another can immediately provide the same service. The task for the network architect is to ensure that a disaster or outage cannot simultaneously impact both the master and all of its slaves. If a disaster occurs, the DNS protocols cannot prevent total loss of name resolution services for hosts within affected zones.

The solution is to disperse name servers in such a way as to avoid single points of failure. At minimum, authoritative name servers for the same zone should be on different network segments so that at least one name server is available in the event that a router or switch fails. This fault tolerance should also extend to wide area data communications lines. For example, if a site has multiple leased lines connecting the network on which the name server resides to a larger network such as the NIPRNet, routing protocols should be configured such that if one of the lines fails, another one will still be available to support the name server.

Organizations should also be prepared for greater disasters, such as the destruction of a building or an entire campus. In cases in which all the hosts defined on an authoritative name server are located in the same building as the name server, then loss of DNS will not impact availability of service because the computing infrastructure is already down. On the other hand, if all the authoritative name servers for a zone reside in a single building, but hosts defined within the zone are located elsewhere, then the loss of the DNS will impact service. The loss of service occurs because users will not be able to resolve host names for servers that are otherwise still operational at an unaffected site.

Since name servers can be dispersed across a network and still be physically located near each other, the DNS architecture should require reasonable geographic dispersal as well. Understandably, many small office sites will not have local name servers. Yet if the host records for a particular site reside on a name server at a remote location, then there should be a backup for that name server at an alternate location. If an organization does not have the resources for this level of dispersal, it can partner with another organization to have each organization's master name server to serve as a slave for the other organization's zone. In this configuration, the name servers are both masters for some zones and slaves for others.

The requirements are as follows:

- *(DNS0200: CAT I) The IAO will ensure that each authoritative master name server has at least one and preferably two or more active slave servers for each of its zones.*

- *(DNS0205: CAT I) The IAO will ensure that not all of the name servers authoritative for a zone are located on the same network segment.*

- *(DNS0210: CAT II) If a zone includes hosts located in more than one building or site, then the IAO will ensure that at least one of the authoritative name servers supporting the zone resides in a different facility from the other name server(s).*

3.4 Authentication and Access Control

The general security objectives for all information systems are *confidentiality*, *integrity*, and *availability*. The primary objective of DNS authentication and access control is the integrity of DNS records – only authorized personnel must be able create and modify resource records; and name servers should only accept updates from authoritative masters for the relevant zones. Integrity is best assured through authentication and access control features within the name server software, though firewalls also can play a significant role in controlling DNS transactions on a network.

A secondary objective of DNS authentication and access control is the confidentiality of DNS records – only those with a valid business purpose should be able to obtain the host names and IP addresses defined within a zone. Both name server configuration and firewalls can support this objective as well. Nevertheless, an organization should never rely on restrictions to DNS host records to provide a significant safeguard to the hosts themselves, as this would constitute a very thin veil of protection. The operating assumption should be that a determined adversary would be able to obtain an IP address of a host within the zone. With a defense-in-depth posture, other controls will prevent the adversary from being able to do much with this address if it were obtained.

3.4.1 Zone Updates

DNS resource records are created and modified through *zone updates*. These occur either through updates of individual records on a master name server or through zone transfers from a master name server to one of its slaves.

3.4.1.1 Updates of Individual Records

Individual resource records can be updated manually – in which case a DNS database administrator either edits the zone file directly or uses a tool to do so – or dynamically, in which case an automated process enters changes to the zone file without the intervention of the DNS database administrator.

The *dynamic update* capability has considerable appeal in an environment in which IP addresses change very frequently – so frequently that it would be unacceptably burdensome or expensive to dedicate the time of a DNS database administrator to this function. This condition would likely be met at sites that rely on the Dynamic Host Configuration Protocol (DHCP) to assign IP addresses to client devices such as workstations, laptops, and IP telephones. It would also apply to sites that utilize frequently changing service (SRV) records.

On the other hand, dynamic updates pose a serious security risk because they can occur without proper authorization and, in some cases, without even the DNS database administrator's knowledge. When dynamic updates are permitted without any compensating controls, any host with network access to the name server can modify any zone record with an appropriately crafted dynamic update request. If an adversary can improperly update records using the dynamic update capability of a name server, this action greatly facilitates a wide variety of subsequent attacks against other machines, thereby placing the entire computing enterprise in jeopardy.

The solution is to require cryptographic authentication of all dynamic update requests, but not all DNS software supports this functionality. When it does not, dynamic updates must be prohibited. BIND 9 name servers support cryptographic authentication of dynamic update requests using Transaction Authentication (TSIG) (a symmetric key technology), but it is relatively rare for DNS clients to support TSIG. As a consequence, dynamic updates are not likely to be feasible for most implementations of BIND. Microsoft's Windows 2000 (and above) DNS supports what it terms *secure dynamic update*. This technology utilizes GSS-TSIG, which leverages a Kerberos-based cryptographic authentication infrastructure shared among both servers and clients. Therefore, Windows 2000 (and above) DNS is effectively the only DNS software that can support dynamic updates for Windows clients.

The requirement is as follows:

- *(DNS0450: CAT I) The DNS software administrator will disable dynamic updates unless the DNS software is configured to require that all dynamic updates be cryptographically authenticated.*

For detailed configuration guidance for dynamic updates within Windows 2000 DNS, refer to *Section 5.3, Secure Dynamic Updates and Active Directory.*

For detailed configuration guidance for dynamic updates within BIND, refer to *Section 4.2.8, The zone Statement.*

3.4.1.2 Zone Transfers

A slave updates its zone information by requesting a zone transfer from its master. In this transaction, the risk for the slave is that the response to its request is not in fact from its authorized master but from an adversary posing as the master. In this scenario, such an adversary would be able to modify and insert records into the slave's zone at will. To protect against this occurrence, the slave must be able to authenticate the master to provide assurance that any zone updates are valid.

The risk to the master in this situation is that it would honor a request from a host that is not an authorized slave, but rather an adversary seeking information about the zone. To protect against this possibility, the master must first have knowledge of what machines are authorized slaves. Then the master must authenticate each slave when the slave requests a zone transfer.

One way an organization can help prevent bogus requests for zone transfers from the master server is to hide the existence of the name server. When this technique is employed, the master

name server is called a *stealth* master. *Hiding* the stealth master simply involves removing its NS record from all related zone files and also ensuring that it does not appear as the primary master name server in any Start of Authority (SOA) records. Slaves are still configured to request zones from the master, but no hosts other than the slaves can learn the master's IP address without access to the software configuration files of the slaves, which should be adequately protected. The stealth master architecture also has the advantage of improving performance because the master is not burdened by client queries and exists only for zone updates and transfers.

The requirements are as follows:

- *(DNS0455: CAT I-II[2]) The DNS software administrator will configure each slave supporting a zone to cryptographically authenticate its master before accepting zone updates.*

- *(DNS0460: CAT III) The DNS software administrator will configure each zone master server to limit zone transfers to a list of active slaves authoritative for that zone.*

- *(DNS0465: CAT III) The DNS software administrator will configure each zone master server to cryptographically authenticate any slave requesting a zone transfer.*

- *(DNS0470: CAT II) The DNS software administrator will configure a name server to only accept notifications of zone changes from a host authoritative for that zone.*

For detailed configuration guidance for zone updates and transfers within BIND 9, refer to *Section 4.2.8, The zone Statement.*

For detailed configuration guidance for zone change notifications within BIND 9, refer to *Section 4.2.6.3, Options Related to Zone Update Notification.*

3.4.2 Query Restrictions

The default configuration in nearly all DNS server software is to permit any client to access any record. There are two reasons why this configuration would be considered insecure. First, some of the records may reveal information about how an internal network is configured and therefore should not be shared with external clients because adversaries could potentially use this information to plan an attack on the internal network. Second, processing queries for any client could allow an adversary with access to the name server to exploit known or unknown flaws in the query process. The exploits might disable the name server, degrade its performance, modify the record cache, or provide the adversary with elevated privileges.

[2] A violation of this requirement can have one of two severity levels depending upon the extent of the violation. If slaves do not authenticate masters in any manner, then the discrepancy would be a Category I finding. If some form of authentication exists (i.e., based on IP address), but it is not based on cryptography, then the discrepancy would be a Category II finding.

3.4.2.1 Split DNS

One way to prevent external clients from obtaining records on internal hosts is to implement *split DNS*. This configuration is most applicable when there is a need to provide authoritative zone information to both internal and external clients, but send different information to each depending on the client's location. For example, external clients (i.e., those outside of an enclave) need resource records on publicly accessible services only (e.g., web site, e-mail, etc.) while internal clients may need resource records on both public services and internal hosts.

The split DNS solution involves separate zone files or views for each type of client. For example, external clients may be directed to the external zone file while internal clients may be directed to the internal zone file. One way to implement this configuration is to host the files on separate name servers. Another method is to host the files on the same server and have two instances of the name server software running, each bound to a separate IP address or interface and then direct internal and external clients to the appropriate IP address. Some DNS software can implement the split DNS capability within the product's configuration. For instance, BIND 9 has this capability through its *named.conf view* statement, which is discussed in more detail in a later section.

The requirement is as follows:

- *(DNS0215: CAT III) When zone records must be accessible from outside an enclave, the IAO will ensure that a split DNS configuration is implemented to prevent records referencing internal hosts from being revealed to external hosts.*

3.4.2.2 Restrictions on Recursion

A potential vulnerability of DNS is that an attacker can *poison* a name server's cache by sending queries that will cause the server to obtain host-to-IP address mappings from bogus name servers that respond with incorrect information. Once a name server has been poisoned, legitimate clients may be directed to non-existent hosts (which constitutes a denial of service) or, worse, hosts that masquerade as legitimate ones to obtain sensitive data or passwords.

To guard against such a scenario, name servers authoritative for .mil domains should be separated functionally from name servers that resolve queries on behalf of internal clients. Organizations may achieve this separation by dedicating machines to each function or, if possible, by running two instances of the name server software on the same machine – one for the authoritative function and the other for the resolving function. In this case, each name server process may be bound to a different IP address or network interface to implement the required segregation.

In addition to enforcing this separation, organizations must ensure that the caching servers only accept queries from known supported clients under the theory that they are much less likely to attempt poisoning attacks than unknown external hosts. In most cases, the hosts inside of an enclave constitute the list of known supported clients. However, in some cases, the caching name server may be expected to serve clients distributed over a wide area, which is acceptable so

long as the name server's audience is still limited in some fashion. In no case, should a name server accept queries from any Internet host.

The requirements are as follows:

- *(DNS0475: CAT III) The DNS software administrator will prohibit recursion on an authoritative name server.*

- *(DNS0480: CAT III) The DNS software administrator will configure a caching name server to accept recursive queries only from the IP addresses and address ranges of known supported clients.*

For detailed configuration guidance for recursion within BIND 9, refer to *Section 4.2.6.2, Options Related to Query and Recursion Restriction.* For guidance related to Windows, refer to *Section 5.5, Forwarders and Recursion.*

3.4.3 Firewalls and DNS

Firewalls are an important component of a defense in depth protection strategy for DNS. One design consideration is the location of name servers relative to the location of firewalls within a network topology. An authoritative name server must never be on the external (or untrusted) side of a firewall. This would in itself make it an untrusted device by definition. Furthermore, it would leave it vulnerable to attack because it would not have the benefit of firewall protection.

Similarly, a name server that handles queries from external hosts must never be on an internal network. Externally accessible name servers must reside within a perimeter network (often called a DMZ) where firewalls can monitor traffic to and from the name servers and the outside world while ensuring that external hosts do not directly communicate with internal hosts. On the other hand, if the name server provides authoritative information exclusively for internal hosts, then it should reside on the internal side of the firewall so it cannot be reached from the outside.

A firewall administrator will need to configure the firewall to support DNS transactions (UDP and TCP 53). The administrator should ensure that this traffic is limited to authorized name servers; in particular, that inbound Port 53 requests to other hosts must be prohibited. The firewall administrator may also need to configure the firewall to support secure shell (TCP 22) or an acceptable alternative means of remote administration of the name server. The exact specification of these rules is beyond the scope of this STIG.

Some organizations use firewalls to limit zone transfers, particularly in cases in which the DNS software does not have this capability. Zone transfers usually involve a TCP connection, while queries usually utilize the connectionless UDP. If, for example, an organization placed an authoritative server in its Internet DMZ but limited zone transfers to servers behind the firewall, then it could allow inbound and outbound UDP 53 to and from the DMZ name server to allow queries, but deny TCP 53 in both directions to prohibit zone transfers. This configuration, however, is strongly discouraged because it may prevent legitimate DNS transactions that involve large responses (e.g., a DNSSEC signature). In general, limitations on queries, zone

updates and transfers should be implemented in the name server's configuration rather than through configuration of firewalls, routers, or other communications devices.

The requirement is as follows:

- *(DNS0405: CAT II) The IAO will ensure that hosts outside an enclave cannot directly query or request a zone transfer from a name server that resides on the internal network (i.e., not in a DMZ).*

3.5 Logging

DNS software administrators need DNS transaction logs for a wide variety of reasons including troubleshooting, intrusion detection, and forensics. These logs should be appropriately secured, having file permissions that restrict unauthorized changes or viewing, and archived, being appropriately backed-up and stored so that they can be examined at a future time. Numerous software products are available to aid the DNS software administrator in examining these transaction logs.[3]

The requirements are as follows:

- *(DNS0485: CAT I-II[4]) The DNS software administrator will configure the DNS software to log, at a minimum, success and failure of the following events:*

  - start and stop of the name server service or daemon
  - zone transfers
  - zone update notifications
  - queries
  - dynamic updates

- *(DNS0490: CAT II) The DNS software administrator will configure the DNS software to send all log data to either the system logging facility (e.g., UNIX syslog or Windows Application Event Log) or an alternative logging facility with security configuration equivalent to or more restrictive than the system logging facility.*

- *(DNS0495: CAT III) The DNS software administrator will configure the DNS software to add timestamps and severity information to each entry in all logs.*

For detailed configuration guidelines for configuring logging within BIND 9, refer to *Section 4.2.5, The logging Statement.*

[3] For a list of tools focused on DNS, visit the ISC web site. From the BIND page, click on "Other BIND and DNS Resources" near the bottom of the page. The URL is http://www.isc.org/products/BIND/contributions.html.

[4] A violation of this requirement can have one of two severity levels depending upon the extent of the violation. If no logging exists, then the discrepancy would be a Category I finding. If some logging exists, but not for all of the events listed, then the discrepancy would be a Category II finding.

For detailed configuration guidelines for configuring logging within Windows 2000 DNS, refer to *Section 5.7, Logging.*

3.6 Zone Files

Many implementations of DNS, including BIND, store zone information in text files, while others store this information in databases primarily accessed through GUI utilities – both methods are acceptable. Regardless of the format, there are basic practices that DNS database administrators should follow when managing the zones for which they have been assigned responsibility.

3.6.1 Change and Ownership Documentation

A zone file should contain adequate documentation that would allow an IAO or newly assigned administrator to quickly learn the scope and structure of that zone. In particular, each record (or related set of records, such as a group of LAN workstations) should be accompanied by a notation of the date the record was created, modified, or validated and record the owner's name, title, and organizational affiliation. The *owner* of a record is an individual with the authority to request that the record be modified or deleted.

This information will help administrators and auditors verify that the zone records are current and that only authorized personnel modify them. If records are not current, there is the potential that an adversary could simulate the activities of a retired host in order to capture logon credentials and other information. For example, suppose a user has a bookmark for a retired web server in his browser. If the record for the server is not removed from DNS, then someone could stand-up another server to mimic the behavior of the retired server, which users may still attempt to access because they may not have deleted or updated the bookmark for that server.

The requirements are as follows:

- *(DNS0220: CAT III) The DNS database administrator will document the owner of each zone record (or group of related records) and the date the record was created, last modified, or verified. This documentation will preferably reside in the zone file itself through comments, but if this is not feasible, the DNS database administrator will maintain a separate database for this purpose.*

- *(DNS0225: CAT II) Record owners will validate their records no less than annually. The DNS database administrator will remove all zone records that have not been validated in over a year.*

3.6.2 Zone-Spanning Records and Glue

If a name server were able to claim authority for a resource record in a domain for which it was not authoritative, this would pose a security risk. In this environment, an adversary could use illicit control of a name server to impact IP address resolution beyond the scope of that name server (i.e., by claiming authority for records outside of that server's zones). Fortunately, all but the oldest versions of BIND and most other DNS implementations do not allow for this behavior. Nevertheless, the best way to eliminate this risk is to eliminate from the zone files any records for hosts in another zone. The two key exceptions to this rule involve glue for NS records and CNAME records for legacy resolution support.

Glue is a term used for a situation in which A records for a delegated zone's name servers appear in the zone file of the parent zone. This is illustrated in the following hypothetical excerpt from the zone file for disa.mil:

```
kalamazoo IN   NS   ns1.kalamazoo.disa.mil
          IN   NS   ns2.kalamazoo.disa.mil
ns1.kalamazoo.disa.mil.  IN   A    132.40.11.2
ns2.kalamazoo.disa.mil.  IN   A    132.40.15.3
```

In this example, the zone file for disa.mil is authoritative for ns1 and ns2 in kalamazoo.disa.mil, a different zone. Yet if the zone file did not include A records for ns1 and ns2, any client seeking records in kalamazoo.disa.mil would be unable to reach that zone. Hence, the glue connecting kalamazoo.disa.mil and disa.mil is a necessary exception to the idea that one zone should not contain authoritative records for another zone.

Another situation in which canonical names from one zone might appear in the zone file of a different zone is in the case of aliases. Suppose the fictitious kalamazoo branch office closed and all of its resources were migrated to the Systems Management Center (SMC) in Oklahoma City. In this situation, the DNS database administrator might retire the kalamazoo.disa.mil domain but still keep its zone operational until the user community has learned of the changes. The DNS database administrator can replace the A records in kalamazoo.disa.mil with CNAME records for the new servers in okc.disa.mil as is shown in the following hypothetical excerpt from the kalamazoo.disa.mil zone file:

```
$ORIGIN kalamazoo.disa.mil
jupiter         IN   CNAME     jupiter.okc.disa.mil.
saturn          IN   CNAME     saturn.okc.disa.mil.
```

In general, zone-spanning aliases should be temporary (e.g., to facilitate a migration). When a host name is an alias for a record in another zone, an adversary has two points of attack – the zone in which the alias is defined and the zone authoritative for the alias's canonical name. This configuration also reduces the speed of client resolution because it requires a second lookup after obtaining the canonical name.

The requirements are as follows:

- *(DNS0230: CAT III) The DNS database administrator will not include any resource records for a host in a zone file if its fully qualified domain name resides in another zone, unless the record is a glue record or temporary CNAME record supporting a system migration.*

- *(DNS0235: CAT IV) The DNS database administrator will ensure that no zone-spanning CNAME record is active for more than six months.*

3.6.3 Improper NS Records and Lame Delegation

A name server (NS) record should map a domain name to an active name server authoritative for that domain. Unfortunately, in poorly configured zone files, these NS records may refer to machines that are no longer in operation or ones that do not provide name services. In some cases, they may provide authoritative name service, but for different zones than the one intended in the NS record. This latter case is called *lame delegation*.

Ill-defined NS records pose a security risk because they create conditions under which an adversary might be able to provide the missing authoritative name services that are improperly specified in the zone file. The adversary could issue bogus responses to queries that clients would accept because they learned of the adversary's name server from a valid authoritative name server, one that need not be compromised for this attack to be successful.

The list of slaves must remain current within 72 hours of any changes to the zone architecture that would affect the list of slaves. If a slave has been retired or is not operational but remains on the list of slaves, then an adversary might have a greater opportunity to impersonate that slave without detection than would be the case if the slave were online. For example, the adversary may be able to spoof the retired slave's IP address without an IP address conflict, which would likely occur if the true slave were active.

The requirement is as follows:

- *(DNS0240: CAT I) The DNS database administrator will ensure that each NS record in a zone file points to an active name server authoritative for the domain specified in that record.*

3.6.4 Root Hints

All caching name servers must be authoritative for the root zone because without this starting point, they would have no knowledge of the DNS infrastructure and thus would be unable to respond to any queries. Of course, being authoritative for a zone means maintaining a zone file for that zone, which often is referred to as the root hints. In nearly all cases, the Internet root hints are part of the installation package of the DNS software. Fortunately, the Internet root servers rarely change their addresses, so DNS software administrators most likely will not have to modify or update root hints.

The security risk is that an adversary could change the root hints and direct the caching name server to a bogus root server. At that point, every query response from that name server is suspect, which would give the adversary substantial control over the network communication of

the name server's clients. The DNS software administrator may use *Appendix G, Root Hints*, to validate the root hints file entries.

A DNS software administrator can obtain root hints from the following locations:

- ftp://ftp.rs.internic.net (198.41.0.6) (for the Internet)
- ftp://www.ssc.smil.mil/ftp/domain/root.cache (for SIPRNET)[5]

DNS software administrators should update root hints periodically (annually should suffice) to ensure that they have current records. There are several methods for obtaining the most current root hints. This file may be obtained from ftp://ftp.rs.internic.net (198.41.0.6) using anonymous File Transfer Protocol (FTP), as mentioned above, to retrieve the *named.root* from the domain directory and then rename this file to the root hints file. The Domain Information Groper (dig) utility is an additional method for obtaining an update of the Internet root hints file. A DNS software administrator can run the following command using the dig utility bundled with BIND and other DNS software:[6]

```
dig @198.41.0.4 . ns > root.hints
```

The requirements are as follows:

- *(DNS0500: CAT 1) The DNS software administrator will ensure that only valid root name servers appear in the local root zone file.*

- *(DNS0505: CAT III) The DNS software administrator will remove the root hints file on an authoritative name server.*

[5] An IP address for this SIPRNET server is not included in the STIG. Retrieving root hints from this URL assumes that one already has a valid root hints file and therefore there is some risk that the user could be directed to a bogus FTP server if this assumption is incorrect.

[6] A similar method for updating the SIPRNET root hints is not included at this time, but may be forthcoming.

This page is intentionally left blank.

4. CONFIGURATION OF BIND

The most common implementation of DNS protocols on Internet and large enterprise name servers is Berkeley Internet Name Domain (BIND). BIND is most often found on UNIX-based operating systems, where it runs as the *named* daemon. There is also an implementation of BIND for Microsoft Windows NT, 2000, and 2003, where it runs as a service.

BIND 9 is the most recent and preferred version of BIND. Some name servers running BIND 8, as well as other alternatives, are still in operation, but they are expected to migrate to BIND 9 soon. There are several advantages of BIND 9 relative to its predecessors:

- Support for DNSSEC
- The ability to differentiate responses to queries using views
- Multiprocessor support
- IPv6 compatibility, including support for IPv6 ports and records
- Improved portability architecture

This section will first explain how to obtain authorized BIND software and then will review security-related configuration of the *named.conf* file, which controls the behavior of BIND.

4.1 Obtaining Software

Systems Administrators should download source code from the Internet Software Consortium (http://www.isc.org). The ISC site is an excellent source for documentation on BIND and its various releases. The administrator should compile the source code using a trustworthy compiler running on a trustworthy computer.

If administrators need to compile code, they should do so on a trustworthy computer other than the name server and then transfer the resulting binaries to the name server for execution. If BIND is compiled on the same machine on which it will eventually run, then this opens the potential for an adversary to either modify the compiler to insert malicious code or use the compiler to create unauthorized programs. In general, no compiler should be resident on a production name server at any time, but it is understood that some versions of UNIX operating systems require local compilers for specific administrative functions.

In some cases, organizations may wish to use operating system vendor supplied versions of BIND in order to comply with or leverage a support agreement with the operating system vendor. This is acceptable so long as the vendor updates its code to be consistent with the most recent versions of BIND in order to avoid security vulnerabilities.

When an SA applies that vendor's patches, the software might be functionally equivalent to a more recent version of BIND than is stated in response to a query. For example, a fully patched version of Solaris 9's BIND might be functionally equivalent to BIND 8.2.7 although a query would respond that it is running BIND 8.2.4. To avoid any confusion, organizations that use vendor-supplied versions of BIND should work closely with the vendor to ascertain consistency with ISC versions of BIND.

An effort is underway to provide a DOD-authorized source for BIND binaries. The network address of this source and when it will become available has not yet been determined. Using a single authorized source will protect DOD against scenarios in which a malicious entity introduces faulty BIND code into the DOD environment to breach security.

The requirement is as follows:

- *(DNS0700: CAT II) The IAO will ensure version BIND 8.2.7 and above, 8.3.4 and above, or 9.2.1 and above is installed. If subsequent IAVA guidance recommends a BIND upgrade, then that guidance will supersede this requirement.*

4.2 Configuration of the named.conf File

Once BIND has been downloaded from the ISC web site and installed, the next step is to configure it. Configuration information is contained in the *named.conf* file, a text file that has special statements and syntax unique to BIND. This section will review the statements that impact security and list requirements for how they should be configured. The relevant *named.conf* statements are as follows:

- key
- controls
- server
- acl
- logging
- options
- view
- zone

For additional information on statements and their syntax, see the *BIND 9 Administrator Reference Manual*. The guide may be obtained from the ISC's web site. On the BIND 9 page (http://www.isc.org/products/BIND/bind9.html), scroll down to the Documentation section and select the "PDF format" link.

4.2.1 The key Statement

The *key* statement defines a key that can be used in Transaction Authentication (TSIG)[7], a technology that authenticates information shared between two or more hosts by signing a hash of transferred information with a private key. Today, TSIG keys will most commonly be used to authenticate that transferred zone data come from an authorized source. Other potential uses include authentication of queries, dynamic updates, and *rndc* control clients. Importantly, the *key* statement does not secure anything by itself. Rather, it labels the keys so they can be used in other statements.

The syntax of the *key* statement is:

```
key key_id {
     algorithm algorithm_name;
     secret string;
};
```

An example of *key* statement in practice might be:

```
key ns1.kalamazoo.disa.mil_ns2.kalamazoo.disa.mil {
     algorithm hmac-md5;
     include "/etc/dns/keys/ns1_ns2.key";
};
```

And the file */etc/dns/keys/ns1_ns2.key* might contain:

```
secret "2njlQNnzn6HTwKLcjStUXg==";
```

In this example the name of the key consists of the two hosts that are going to share the key, namely the name servers ns1 and ns2 in the fictitious domain kalamazoo.disa.mil. The algorithm is the hmac-md5 hashing algorithm, the only one currently supported in BIND 9. The use of the *include* statement nested within the *key* statement allows the DNS software administrator to separate the *named.conf* file from the actual key strings. This is helpful when there is a requirement to hide the keys from those who have the ability to view the *named.conf* file.

The generation of the required keys and their placement in the correct files involves several steps. Administrators should use the key generation tools supplied with the DNS software and not third party utilities. Administrators who are not familiar with this process may consult *Appendix C, Generation and Storage of TSIG Keys,* for further information.

[7] Transaction Authentication was previously referred to as Transaction Signature (abbreviated as TSIG), and the old term may still be used in many references. When the name changed, the TSIG abbreviation remained.

A future release of BIND 9 is expected to include support for *SHA-1*, which is the algorithm currently specified in the National Institute of Standards and Technology's (NIST's) Secure Hashing Standard (FIPS 180-1) and required throughout DOD. When it is available, hmac-md5 should be replaced with hmac-sha1 for DNS TSIG applications. In general, only NIST or National Security Agency (NSA) approved algorithms should be utilized in the DOD computing infrastructure.

The requirements are as follows:

- *(DNS0705: CAT III) The DNS software administrator will utilize 128-bit HMAC-SHA1 keys when available and 128-bit HMAC-MD5 keys until that time.*

- *(DNS0710: CAT II) The DNS software administrator will place each required key in its own dedicated file.*

4.2.2 The controls Statement and the rndc.conf File

The *controls* statement defines how BIND will listen for control messages (e.g., commands that reload zones, stop the name server, and dump a copy of the name server's internal database to a file). A DNS software administrator can use the BIND 9 *rndc* utility to generate the control messages. The BIND 8 equivalent is the *ndc* utility.

The risk posed by the *controls* statement and the associated use of the *rndc* or *ndc* commands is that an adversary could use them to remotely control the name server without having to authenticate to the operating system on which the name server resides. To prevent this occurrence, one can limit control messages to the local host, thereby prohibiting the use of the control utilities from remote hosts. This restriction should not hinder administration because authorized DNS software administrators should have user accounts on the machine and be able to remotely access it. Therefore, they can log on remotely and then run the control utility locally.

If a DNS software administrator would like to utilize the BIND 9 *rndc* statement under the restrictions just described, he or she can execute the command **rndc-confgen -a** and apply appropriate permissions (i.e., 440) to the resulting *rndc-key* file that should appear in the */etc* directory. The default configuration that results from this procedure will meet all security requirements. The remainder of this section provides additional information in case more complex configurations are required.

The syntax of the *controls* statement is:

```
controls {
     inet ip_address | *
          [ port ip_port ]
          allow { address_match_list }
          keys { key_list };
     [ inet ... ]
};
```

An *inet* phrase will exist for the network interface(s) on the name server that listens for control messages. One would enter "*" to enable control messages to be heard on any interface. The listening port is configurable but defaults to TCP 953. The *allow* and *keys* statements provide security – the listener will accept messages only from those IP addresses listed in the *allow* clause and only when accompanied by a key listed in the key list.

A corresponding *key* statement in the *named.conf* file must accompany the *keys* phrase. The previous section reviewed the syntax for the *controls* statement and related security requirements.

An example of the *controls* statement in practice is:

```
controls {
     inet 127.0.0.1
     allow 127.0.0.1
     keys { "rndc_key" };
};
```

To complete the configuration, the *named.conf* file would also include a *key* statement for the *rndc_key*.

The statement above restricts the use of *rndc* to the local host, which is the desired behavior. If there is no *controls* statement in the *named.conf* file, *named* will implement this behavior by default and look for *rndc_key* in */etc/rndc.key* (or the path defined using the sysconfdir parameter when the BIND build was configured).

rndc also has its own configuration file, *rndc.conf*, that has a similar syntax to the *named.conf* file, but is limited to the *options*, *key*, *server*, and *include* statements. An example of a minimal configuration is as follows:

```
key rndc_key {
     algorithm hmac-md5;
     secret "2njlQNnzn6HTwKLcjStUXg==";
};
options {
     default-server localhost;
     default-key rndc_key;
};
```

A DNS software administrator can run **rndc-confgen** to create this file and a corresponding suggested *controls* statement for the *named.conf* file.

The requirement is as follows:

- *(DNS0715: CAT II) The DNS software administrator will configure BIND to accept control messages only when the control messages are cryptographically authenticated and sent from an explicitly defined list of DNS administrator workstations.*

4.2.3 The server Statement

If a *key* statement defines a key intended to support TSIG, then it must be combined with a corresponding *server* statement to produce the desired behavior. The relevant syntax of the *server* statement is:

```
server ip_address {
     keys {string; [string; […]]};
};
```

In our example, suppose ns1.kalamazoo.disa.mil has IP address 10.1.1.1 and ns2.kalamazoo.disa.mil has IP address 10.2.2.2. In this case, the following statement referencing ns2 would appear on ns1:

```
server 10.2.2.2 {
     keys {ns1.kalamazoo.disa.mil_ns2.kalamazoo.disa.mil};
};
```

Likewise, the reciprocal statement referencing ns1 would appear on ns2:

```
server 10.1.1.1 {
     keys {ns1.kalamazoo.disa.mil_ns2.kalamazoo.disa.mil};
};
```

Each name server will sign responses with the specified key, but will not require that requests be signed. Hence, both servers must include the relevant server statement to ensure that both ends of a transaction are signed. Note that both servers use the same key for communication between them.

The requirement is as follows:

- *(DNS0720: CAT II) The IAO will either ensure that any two name servers sharing zone information utilize a unique TSIG key for communication between them or, in cases in which more than four servers support a zone, create a written key management plan that will document how keys are shared and replaced in a manner to reduce residual risk to an acceptable level.*

4.2.4 The acl Statement

The *acl* statement implements Access Control Lists (ACLs) within the *named.conf* file. It assigns a name to a list of IP addresses, networks, or keys to simplify references in other statements. For example, it is easier to read the name trusted_hosts or internal than a long list of subnets. BIND ACLs should not be confused with access control lists on routers or server files, which must also be modified to secure DNS. In the case of BIND, the ACLs merely label various hosts and networks for use in later statements.

The syntax of the *acl* statement is:

```
acl acl-name {
     address_match_list
};
```

An example of *acl* for IP hosts inside an enclave might be:

```
acl trusted_hosts {
     225.10.230.0/24;
     239.10.241.128/25;
     239.10.241.15;
};
```

In this example, the list of trusted hosts consists of the networks 225.10.230.0/24 and 239.10.241.128/25, and a host with IP address 239.10.241.15.

ACLs may also include TSIG keys, as in the following example in which ns1 communicates with ns2 and ns3:

```
acl zone_partners {
     "ns1.kalamzoo.disa.mil_ns2.kalamazoo.disa.mil";
     "ns1.kalamzoo.disa.mil_ns3.kalamazoo.disa.mil";
};
```

In this example, ns1 only accepts responses from ns2 and ns3 when they are authenticated with the appropriate TSIG key.

As a best practice, DNS software administrators should create BIND ACLs for all supported hosts or networks that will be permitted to use the name server on which the configuration file resides. The DNS software administrator should also create an ACL for the name servers with which it will transfer zone information. Organizing the relevant hosts and networks in this fashion will simplify the administration and audit of the *named.conf* file.

4.2.5 The logging Statement

The default behavior for BIND 8 and 9 complies with the general logging requirements specified in *Section 3.5, Logging*. In many cases, the default configuration may not be appropriate for

operational reasons. For example, separation of duties between SAs and DNS software administrators may require them to maintain separate logs, perhaps one capturing events that the other does not. The precise nature of the logging configuration is at the discretion of the site so long as it meets the general requirements. Organizations that desire non-default logging behavior will need a strong understanding of the *logging* statement and its *categories* and *channels* phrases, which are explained in the remainder of this section.

The *logging* statement controls what BIND messages will be sent and to where. The *categories* substatement (or phrase) defines the what part of logging – i.e., the scope and content of log entries. The *channels* phrase defines the where part of logging – i.e., the file or process to which the log entries are written.

The syntax of the *logging* statement is:

```
logging {
     [ channel channel_name
          ( file path_name
               [ versions ( number | unlimited ) ]
               [ size size_spec ]
            | syslog syslog_facility
            | stderr
            | null );
          [ severity (critical | error | warning | notice |
               info | debug [level]| dynamic); ]
          [ print-category yes/no; ]
          [ print-severity yes/no; ]
          [ print-time yes/no; ]
     }; ]
     [ category category_name {
          channel_name ; [ channel_name ; … ]
     }; ]
};
```

An example of the *logging* statement in practice might be:

```
logging {
     channel "query_log" {
          file "/var/dns/named.query"
               versions unlimited
               size 30 m;
          severity info;
          print-category no;
          print-severity yes;
          print-time yes;
          };
     category "queries" {
          "query_log";
          };
};
```

This statement will log DNS queries to a file */var/dns/named.query*. Entries will include the severity of the message and the time and date it occurred, but will not include the category name (we already know that it is a query). When the log size reaches 30 MB, a new log will be opened. Query logs can grow very quickly, so setting limits on their size ensures that they can be managed appropriately (e.g., moved to CD-ROM for archival, viewed in a text editor, etc.). In addition, if a size limit is set, unusually rapid growth in the number of query logs is a good indication that there is something wrong on the network and perhaps a denial of service attack is in progress. When selecting the log size for a particular name server, DNS and System Administrators should consider the server's disk space utilization, the typical volume of DNS transactions, and the expected log rotation and archival scheme. Lack of disk space can be problematic for server performance and availability.

The rationale for logging query data to a separate log is that in most environments, the number of query entries is likely to greatly exceed the entries of all other categories combined. Important messages are more likely to be overlooked during daily log reviews if they are easily lost in the noise. Moreover, archival requirements for system information will exceed those of query information in nearly all cases. Combining both types of information into a single log would require a great deal of data to be stored much longer than necessary, which increases administrative burden and cost.

The *logging* statement will also log most other behavior via a category named *default*, which does not need to be explicit in the *named.conf* file. There are several predefined phrases that the DNS software administrator should know to fully understand the logging behavior of BIND. The most relevant predefined phrases are:

```
channel "default_syslog" {
     syslog daemon;
     severity notice;
};
channel "default_debug" {
     file "named.run";
     severity dynamic;
};
category "default" {
     "default_syslog";
     "default_debug";
};
```

If the DNS software administrator needs to supplement these predefined phrases to meet a business requirement, then the complete phrase needs to be included in the *named.conf* file. The DNS software administrator should consult the *BIND 9 Administrator Reference Manual* for more detailed information on the *logging* statement and its *channel* and *category* phrases. The information here constitutes only a minimally acceptable configuration for BIND logging.

On UNIX systems, the DNS software administrator should also read the syslog man pages to determine how *syslog.conf* statements may interact with the logging directives in the *named.conf* file (simply type **man syslog** at the command prompt to obtain this information). In general, the *syslog.conf* file settings will dominate the *named.conf* file settings if they are set at a higher severity level. For example, if one specifies that a BIND logging channel records everything of severity notice and above, but syslog is configured to log only critical messages, then only critical messages will be captured. The DNS software administrator may need to modify one or both of the *named.conf* and *syslog.conf* files to ensure that STIG requirements are met.

In the example above, there is a special *channel* for the queries *category* and all other logging and categories log to the default *channel* and default *category*. Therefore, entries of severity level notice and above will be sent to the syslog daemon for all categories other than the query category. The *named.run* file will capture all entries other than queries at the global severity level of the operating system (which results from using the value *dynamic* for the severity parameter). The DNS software administrator should maintain a separate log for the queries category to avoid crowding critical application messages with extensive query records. In addition, the DNS software administrator should set query log size restrictions appropriate to the environment, but never to exceed 80 percent of available disk space.

4.2.6 The options Statement

The *options* statement in the *named.conf* file defines global options for BIND. Some options can also be defined for each zone. Not all options impact security. The relevant BIND options for this STIG are as follows:

- version
- recursion
- forward
- forwarders
- allow-query
- allow-transfer
- allow-recursion
- notify
- allow-notify
- also-notify
- fake-iquery
- rfc2308-type1

The general syntax of the *options* statement is:

```
options {
     [ option1 option_value1; ]
     [ … ]
     [ optionN option_valueN; ]
};
```

Importantly, the *options* statement on an authoritative name server will look quite different than one on a caching name server because of the different roles these servers have in the DNS infrastructure. The following is an example of an *options* statement that includes security-relevant parameters for a DNS STIG compliant configuration on an authoritative server:

```
options {
     version " ";
     recursion no;
     allow-query {none;};
     allow-transfer {none;};
};
```

Again, there will be other options listed in a typical *options* statement. Only security-relevant phrases are shown above. Each of these options will be discussed in more detail later in the section.

In the example above, both queries and transfers are prohibited. At first, these prohibitions might seem strange since obviously DNS requires both queries and transfers to function properly. The *allow-query* and *allow-transfer* phrases, however, can appear in both the *options* statement and the *zone* statement. Any settings in the *zone* statement override any included in

the *options* statement. If the *zone* statement contains no such settings, then the settings in the *options* statement are utilized. Given the interaction between the two statements with respect to the *allow-query* and *allow-transfer* phrases, the most secure configuration is to deny everything in the *options* statement and then permit only what is necessary in the *zone* statements. The desired behavior when a zone has been incompletely configured is to prevent any transactions from occurring.

On a caching server, a DNS STIG compliant configuration might include the following:

```
options {
     version " ";
     recursion yes;
     forward "first";
     forwarders {forwarding_servers;};
     allow-query {trusted_hosts;};
     allow-recursion {trusted_hosts;};
};
```

The major differences between the authoritative and caching name server configurations are as follows:

- Caching servers allow recursion and forwarding.
- Caching servers only accept queries from defined hosts.

Note that the BIND ACLs "forwarding servers" and "trusted hosts" are utilized here. The alternative is to list the IP addresses and address ranges in the *option* phrase. Using ACLs is the preferred approach because it helps prevent against an attack in which an errant host is inserted into a potentially long lists without detection. With ACLs, auditors and administrators need only examine the *acl* statements to verify the integrity of referenced hosts within the *named.conf* file. The use of ACLs also makes the options statements easier to read.

The remainder of this section discusses each of the options in more detail.

4.2.6.1 The version Option

The *version* phrase in the *options* statement determines how the name server will respond to queries regarding which version of BIND is running. Responding with this information could notify the attacker of what potential vulnerabilities might exist on the name server. To protect against this scenario, we can configure BIND to provide a null response using the following phrase:

```
options {
     version " ";
     [ … ]
};
```

Alternative text is acceptable so long as it does not include version information, as in the following example:

```
version "Version not disclosed";
```

The requirement is as follows:

- *(DNS0725: CAT III) The DNS software administrator will configure BIND not to disclose its version number to queries.*

4.2.6.2 Options Related to Query and Recursion Restrictions

A potential vulnerability of DNS is that an attacker can poison a name server's cache by sending queries that will cause the server to obtain host-to-IP address mappings from bogus name servers that respond with incorrect information. Once a name server has been poisoned, legitimate clients may be directed to non-existent hosts (which constitutes a denial of service) or, worse, hosts that masquerade as legitimate ones to obtain sensitive data or passwords.

To guard against such a scenario, name servers authoritative for .mil domains must be separated functionally from name servers that resolve queries on behalf of internal clients. In addition to enforcing this separation, organizations must ensure that the caching servers only accept queries from trusted clients under the theory that they are much less likely to attempt poisoning attacks than unknown external hosts. These general requirements were first stated in *Section 3.4.2.2, Restrictions on Recursion*.

The following BIND options can implement the desired DNS architecture:

- recursion
- forward
- forwarders
- allow-query
- allow-recursion

On an authoritative server, if the DNS software administrator sets *recursion* to "no" and leaves *forward* undefined, then poisoning will be prevented because the name server will never obtain host-to-IP address mappings from external hosts. As discussed above, the *allow-query* option appears in both the *options* statement and the *zone* statement. To keep the default behavior as restrictive as possible, the DNS software administrator sets the *allow-query* option to "none" and permits queries as required in the *zone* statement. With regards to the *options* statement, these objectives are accomplished with:

```
options {
     [ … ]
     recursion no;
     allow-query {none;};
     [ … ]
};
```

The "recursion no;" phrase implements the first general requirement listed in *Section 3.4.2.2, Restrictions on Recursion,* which prohibits recursion on an authoritative name server.

On a caching server, the DNS software administrator wants to permit recursion and forwarding, but limit it to trusted hosts. In the *named.conf* file, this is accomplished with:

```
options {
     […]
     recursion yes;
     forward "first";
     forwarders {forwarding_servers;};
     allow-query {trusted_hosts;};
     allow-recursion {trusted_hosts;};
     […]
};
```

The "allow recursion {trusted_hosts;}" phrase implements the second general requirement listed in *Section 3.4.2.2, Restrictions on Recursion,* which allows caching servers to accept recursive queries only from the IP addresses and address ranges of known supported clients.

The terms "forwarding_servers" and "trusted_hosts" in these options are BIND ACLs defined with the *acl* statement. As mentioned, the use of these ACLs makes the *named.conf* file easier to read, manage, and audit.

Another approach to limiting queries is to use the *option* statement's *blackhole* phrase, which forces the name server to drop any query from a host whose source address matches the phrases address match list. In a sense, the *blackhole* phrase is the inverse of the *allow-query* phrase, specifying a list of hosts from which queries are prohibited rather accepted, as is the case with *allow-query*. For example, if a site does not use RFC 1918 private address space (e.g., 10.0.0.0 network addresses), then any request coming from such an address would be illegitimate and should be dropped. In other cases, there may be known address ranges from which DNS attacks are likely. Despite these situations, use of the *blackhole* phrase is not recommended because these types of address restrictions are more efficiently implemented and easier to maintain in firewall rules and router ACLs. DNS software administrators should not be expected to frequently update the *blackhole* list based on emerging and rapidly changing threats.

4.2.6.3 Options Related to Zone Update Notification

Section 3.4.1.2, Zone Transfers lists a general requirement that name servers only accept notifications of zone changes from the hosts authoritative for that zone. If all of a zone's NS records are valid, then the default behavior in BIND complies with this requirement and does not require the DNS software administrator to take any additional action.

In some cases, the DNS software administrator must implement a non-default configuration to comply with operations requirements. If this is the case, the DNS software administrator must have an understanding of the *named.conf* options that govern how master name servers notify

other hosts of zone changes and when slave servers will accept notifications. If none of these options are selected, the resulting behavior represents an acceptable security risk.

The three phrases within the *options* statement that govern this behavior are:

- *notify* – which turns notification on or off (defaults to on)

- *also-notify* – which defines servers other than those listed in NS records that will be sent notifications (defaults to none)

- *allow-notify* – which defines from which servers a slave will accept notifications (defaults to the master name server only)

In summary, the default behavior is as follows:

- A master name server sends notifications to all other authoritative servers in a given zone (as defined by NS records for that zone) whenever the zone changes.

- A slave name server only accepts notification of zone changes from its master.

A requirement for non-default behavior is uncommon. Some sites may have a *stealth slave* – i.e., a name server for which there is no NS record in the zone, but which nonetheless is intended to be authoritative for that zone. If a stealth slave does not appear in an *also-notify* phrase in the *options* statement, it would not receive notifications but it still could periodically pull zones without notifications.

4.2.6.4 IQUERY Simulation

In earlier versions of BIND, there is a security vulnerability associated with simulating the obsolete query type IQUERY, which was implemented with the *fake-iquery* phrase in the options statement. BIND 9 never does IQUERY simulation.

The requirement is as follows:

- *(DNS0730: CAT III) The DNS software administrator will ensure that the named.conf options statement does not include the phrase "fake-iquery yes;".*

4.2.6.5 RFC 2308

A potential vulnerability in DNS is that a name server will provide a list of name servers (i.e., NS records) along with negative responses to queries. Presumably, this response is intended to provide additional assistance to a client trying to resolve a host name to an IP addresses. The general idea is, "I cannot help you but here are lists of servers I know that might be able to." Unfortunately, if the query were not sent from a legitimate lost client but an attacker attempting to infiltrate a network, then the name server would provide valuable information to that attacker.

To ensure that this undesirable behavior is not supported on a given name server, the DNS software administrator must set the option *rfc2308-type1* to "no", which is the default behavior in BIND 9.

The requirement is as follows:

- *(DNS0735: CAT II) The DNS software administrator will ensure that named.conf options statement does not include the phrase "rfc2308-type1 yes;".*

4.2.7 The view Statement and Split DNS

To maintain the confidentiality of information about a network, a DNS software administrator may want to restrict untrusted outsiders from learning the names and IP addresses of hosts within the internal network while also allowing external hosts to reach public web sites, e-mail servers, and other services supporting the organization's mission. The solution is to *split* the DNS into two: a version accessible to outsiders that contains limited information and another version for internal use that contains all information.

In the past a split DNS was achieved by hosting different zone data on different servers – one residing on a publicly accessible DMZ and another on the internal network. Another complex but acceptable alternative is to run two instances of BIND on the same server, each bound to a unique IP address and loading a distinct zone file. This mechanism for achieving split DNS is sometimes referred to as *split brain* DNS.

BIND 9 introduces the ability to generate *views* on zones that can direct clients to different records based on the source IP address of the client generating the query. This greatly simplifies the administration of a split DNS and can also reduce the number of name servers required to support the DNS infrastructure.

Views are implemented using the *view* statement in the *named.conf* file. The syntax of the *view* statement is as follows:

```
view view_name {
     match-clients { address_match_list };
     match-destinations { address_match_list };
     recursion yes/no;
     [ view_option; … ]
     [ zone-statistics yes/no; ]
     [ zone_statement; … ]
};
```

An example of *view* statements in practice might be as follows:

```
view "internal" {
     match-clients {trusted_hosts;};
     recursion no;
     zone kalamazoo.disa.mil {
```

```
            type master;
            file "kalamazoo-internal.db";
        };
    };
    view "external" {
        match-clients {any;};
        match-destinations {dmz;};
        recursion no;
        zone kalamazoo.disa.mil {
            type master;
            file "kalamazoo-external.db";
        };
    };
```

First, note that the *zone* statements in the example above only contain the minimum information to support views and, therefore, are incomplete security configurations. The next section provides a more complete discussion of the *zone* statement.

The ordering of the views is often critical because the source and destination IP addresses of a query are matched against each view in the order they appear in the *named.conf file*. Once a query matches one of the views, no further views are checked. As a result, the DNS software administrator must be cognizant of potential problems that might result from view sequencing.

In this example, the internal view precedes the external view. Consequently, internal clients are always served by the internal view because the phrase "match-clients {trusted_hosts;};" would always be true in this circumstance. Alternatively, had the external view preceded the internal view, internal clients would be captured in the phrase "match-clients {any;};" included in the external view. Thus, the internal clients would utilize the external view to resolve names of hosts in the DMZ. In this case, the ordering does not impact DNS behavior, but in more complex cases, it might.

The site may implement a split DNS in whichever way it deems appropriate so long as it meets the requirement in *Section 3.4.2.1, Split DNS,* that records related to internal hosts not be revealed to external clients.

4.2.8 The zone Statement

Zone configuration is achieved with the *named.conf zone* statement. The *zone* statements dictate query handling, permissible zone transfers, the location of zone data and more. They tie together and make sense of the *acl, key, server,* and *options* statements that precede them.

The *zone* statements come with their *zone* options, which are similar to the global *named.conf* options. Recall that zone options supersede global options. The general strategy is to make the global options as restrictive as possible and then loosen these restrictions in *zone* statements based on business requirements. This strategy follows the security principle of *least privilege.*

The *zone* statements can either appear on their own within the *named.conf* file or be nested within *view* statements. However, in the presence of a *view* statement, a standalone *zone* statement would be rendered superfluous – it would never be utilized by any view and, consequently, by any client. As a result, if any *view* statements are explicit in the *named.conf* file, then all *zone* statements must be nested within *view* statements. Most *zone* statements in the DOD computing environment will be located within *view* statements given the requirement for split DNS.

The syntax of security-relevant components of the *zone* statement is as follows:

```
zone zone_name {
     type ( master | slave | hint | stub | forward );
     allow-query { address_match_list };
     allow-transfer {
          address_match_list;
          key key_ID
     };
     allow-update {address_match_list };
     file string;
};
```

The general security requirements related to the *zone* statement, which are listed in *Section 3.4.1.2, Zone Transfers,* state that zone transfers must be limited authorized name servers and that these transfers be cryptographically authenticated. An example STIG-compliant *zone* statement in practice might be as follows:

```
zone kalamazoo.disa.mil {
     type master;
     allow-query {any;};
     allow-transfer {
       key ns1.kalamazoo.disa.mil_ns2.kalamazoo.disa.mil;
     };
     allow-update {none;};
     file "kalamazoo.db";
};
```

In the example, the phrase "allow-query {any;};" permits DNS queries from any host, which is the default behavior and the appropriate configuration for an authoritative server. Recall, however, that in the *named.conf options* statement the default was changed to "allow-query {none;};" (i.e., deny all queries unless explicitly told otherwise). Accordingly, queries must be activated in the *zone* statement for it to function properly.

In the example, the use of a key in the *allow-transfer* phrase activates transaction signatures (or TSIG) that ensure that transactions are authenticated. The slave server ns2.kalamazoo.disa.mil is the only other server that has this key, so this phrase ensures that zone transfers are restricted to the slave only.

In the example, the phrase "allow-update {none;};" disables dynamic updates. Dynamic updates are useful when hosts frequently change their IP address (e.g., workstations on a local area network utilizing DHCP). Unfortunately, dynamic updates represent a very serious security vulnerability because they allow an attacker to modify zone records using bogus updates from spoofed addresses. Therefore, dynamic updates must be cryptographically authenticated if they are permitted.

In the example provided above, the *zone* statement restricted zone transfers to those signed by a particular TSIG key. While cryptographic authentication satisfies the general requirements in this STIG, in a maximum-security posture zone transfers and dynamic updates are restricted both by IP address and by TSIG key. Such a configuration would provide additional protection against a scenario in which someone is able to obtain the key (e.g., via theft) and then use it from an unauthorized host, because the adversary both would have to obtain the TSIG key and spoof the IP address of the slave name server.

The implementation of this configuration is somewhat tricky because it utilizes the negated match feature twice – first in an *acl* statement and a second time within the *allow-transfer* statement. Nevertheless, this configuration is recommended. An example is as follows:

```
acl not_ns2.kalamazoo.disa.mil {!239.10.241.129; any; }

zone kalamazoo.disa.mil {
   …
     allow-transfer { !not_ns2.kalamazoo.disa.mil;
       key ns1.kalamazoo.disa.mil_ns2.kalamazoo.disa.mil;
     };
   …
};
```

In this example, the IP address of the ns2.kalamazoo.disa.mil is 239.10.241.129. The *acl* statement defines an access list that includes every host except the ns2. The “!” character before any list entry means not – in this example, not 239.10.241.129. If a negation element is true, BIND moves to the next element in the list – in this example, “any”.

The negation operator in the *allow-transfer* statement, the first element is effectively a double negative. It is true when the requesting host is 239.10.241.129 and false when it is not. Recall, however, that BIND continues to check other items in the list when a negation statement is true. Therefore, through the use of the double negative, the requesting host must both have IP address 239.10.241.129 and meet at least one other condition in the access list. In this case, the other condition is that the host signed its request with the correct TSIG key. If either condition is not met, then access is denied. DNS software administrators should secure zone transfers and dynamic updates using this technique when it is feasible to do so.

5. CONFIGURATION OF WINDOWS 2000 DNS

DISA and many other DOD organizations utilize Windows 2000 or 2003 Server[8] technology to support their Windows networks. Windows 2000 is heavily dependent on DNS and comes bundled with its own proprietary DNS software. Unfortunately, Windows 2000 DNS does not support several important features. Therefore, personnel who support Windows 2000 DNS should prepare for a forthcoming mandate to migrate to BIND 9 or other DOD-mandated alternative.

This section provides interim guidance for Windows 2000 DNS implementations in use prior to the migration to BIND or other DOD-mandated alternative.

5.1 Comparison of Windows 2000 DNS and BIND

To understand the security posture of Windows 2000 DNS relative to BIND 9, one must know the features that each of the DNS software suites support. There are some features supported in BIND 9 that are not supported in Windows 2000 DNS. Similarly, there are features in Windows 2000 DNS that are not currently supported in BIND 9.

5.1.1 BIND 9 Security Features Not Present in Windows 2000 DNS

BIND 9 has the following features that Windows 2000 DNS does not have:

- TSIG support for zone transfers and queries
- DNSSEC support, especially the ability to cryptographically sign zones
- Ability to restrict queries and recursion by IP address
- Ability to explicitly define split DNS configurations on an authoritative name server

TSIG is required for zone transfers, but not for queries. Zone transfers that occur without the cryptographic authentication that TSIG provides are less secure than zone transfers that rely on other forms of authentication such as source IP address. Reliance on weak authentication methods for zone transfers increases the possibility that adversaries can alter authoritative records, which has significant implications for the integrity of the DOD DNS infrastructure. Moreover, if an adversary is able to perform a zone transfer, he or she can map many network resources with a single transaction, which is much less likely to be detected than an authorized scan of the network.

DNSSEC provides cryptographic authentication of query responses through the use of signed zones. BIND 9 name servers can check the validity of signatures on behalf of clients, while Windows 2000 DNS servers have no ability to do this. This is a major shortcoming of Windows 2000 DNS because DNSSEC signed zones are a cornerstone of future efforts to secure DOD DNS transactions.

[8] As mentioned in the introduction, Windows 2000 and 2003 DNS are very similar and have identical security configuration requirements. Throughout this document, references to Windows 2000 will also apply to Windows 2003.

Windows 2000 is able to restrict zone transfers by IP address; however, unlike BIND 9, cannot restrict queries and recursion by IP address. As mentioned, IP address restrictions are a weak form of authentication for zone transfers, but they currently are the only practical form of restricting transactions that involve direct interaction between name servers and DNS client resolvers. Without IP address restrictions, sensitive host records may be sent to an unauthorized user. In addition, the potential exists for adversaries to poison the name server's cache if it will accept their queries.

Address restrictions are particularly important in the implementation of split DNS, in which the objective is to provide different host information to different clients depending on their network location. With Windows 2000, the only means to create a split DNS is to use two independent name servers authoritative for the same zone and then use firewall rules or router ACLs to restrict access to those name servers by network address. This is a highly artificial and undesirable architecture. In BIND 9, one can use views to distinguish what information will be given to whom.

5.1.2 Windows 2000 Security Features Not Present in BIND 9

The notable advantages of Windows 2000 DNS relative to BIND 9 are as follows:

- Support for (Generic Security Service) GSS-TSIG
- Granular access rights to DNS zones and objects

The key feature in Windows 2000 DNS that is absent in BIND 9 is GSS-TSIG, which is the basis for the Windows 2000 DNS "Secure Dynamic Updates" feature. BIND 9 can be configured to accept dynamic updates, but can only authenticate them through TSIG shared secret keys. Very few clients support TSIG today, and even if they did, supporting such clients in a large environment would impose an unacceptable administrative burden given the large number of secret keys the system would require.

GSS-TSIG differs from ordinary TSIG in that it leverages an existing security relationship for authentication and session keying. In the case of Windows 2000, DNS can be integrated with Microsoft's Active Directory (AD), a distributed database of user and machine information and other records. AD uses the accounts associated with computers defined within AD to provide the existing security relationship for GSS-TSIG. This eliminates the administrative burden of managing separate TSIG keys.

With the use of GSS-TSIG, Windows 2000 offers reasonable (but not absolute) assurance that dynamic updates are legitimate – a powerful advantage when a network supports DHCP clients, whose IP addresses may change frequently. If an organization has a requirement for DHCP clients to serve data or an application to other users, then secure dynamic updates offer the only feasible means to operate under those conditions. The alternative is for the DNS database administrator to manually update the zone records whenever a DHCP client obtains a new IP address, which is an unreasonably burdensome approach.
The second advantage of Windows 2000 DNS relative to BIND 9 is that AD integration allows an administrator to grant user groups AD permissions to DNS zones and objects. This feature offers far more granular control of DNS objects than BIND does. For example, a System

Administrator can assign various other DNS software administrators different levels of access to each zone depending upon their roles and responsibilities. In BIND, on the other hand, a System Administrator cannot grant other administrators write permissions to particular sections of the *named.conf* file, where options for all zones are defined. In BIND, an administrator either has full control or no control.

Finally, Windows 2000 requires a number of SRV records in the DNS for the LDAP, Kerberos, and catalog services that are critical to the operation of Windows 2000. Without these services, users could not log on, obtain directory information or perform other critical functions. When Windows 2000 servers utilize Windows 2000 DNS, these records are automatically entered into the zone upon installation and automatically modified later when there are changes. This feature is desirable because the SRV entries can be intimidating for administrators new to either Windows 2000 or DNS in general. In fact, the transfer of these records from Windows 2000 DNS to BIND 9 is not difficult. They can be simply cut and pasted from a Windows DNS zone into a BIND 9 zone. Changes to the SRV records would require manual intervention, but this is not expected to be a frequent occurrence on most networks.

5.1.3 Implications of BIND 9/Windows 2000 DNS Differences

DISA has a strong preference that organizations deploy BIND 9 to support DNS for the following reasons:

- BIND 9 supports several key required security features that Windows 2000 DNS and most other DNS products do not.

- DISA examines, compiles, and rigorously tests BIND 9 code but does not have similar access to Windows 2000 DNS code nor does it perform similar evaluation of that code.

- DISA has a large investment in BIND 9 software development and a significant voice in the future direction of BIND, while it has less influence over Microsoft's future DNS development effort.

At the same time, Windows 2000 has certain features that make it an acceptable interim solution for many environments. The use of Windows 2000 DNS should be limited in such a manner to reduce the cost and complications of the future migration to BIND. For instance, there is no reason to use Windows 2000 DNS as a caching name server or for any application that requires recursion. Microsoft Windows clients can still send queries to BIND 9 name servers, which can in turn query authoritative Windows 2000 DNS servers when needed.

GSS-TSIG support is expected to be available and tested in BIND 9 in Q1 2004. Once GSS-TSIG is available in BIND 9, there should no longer be a need to deploy Windows 2000 DNS. At that point, BIND 9 will provide all the services and features that make Windows 2000 DNS an attractive alternative for some applications today. Of course, both software functionality and security requirements can change rapidly so DOD guidance on this issue may change as well.

The remainder of this section provides specific implementation guidance for organizations that still choose to deploy Windows 2000 DNS.

5.2 Network Restrictions

As mentioned, Windows 2000 DNS cannot restrict transactions other than zone transfers by IP address. Instead, organizations must use firewalls or router ACLs to enforce those restrictions. These additional protections are typically a component of defense-in-depth security architecture, but in this case they comprise the first and likely only line of defense against unauthorized access to zone records.

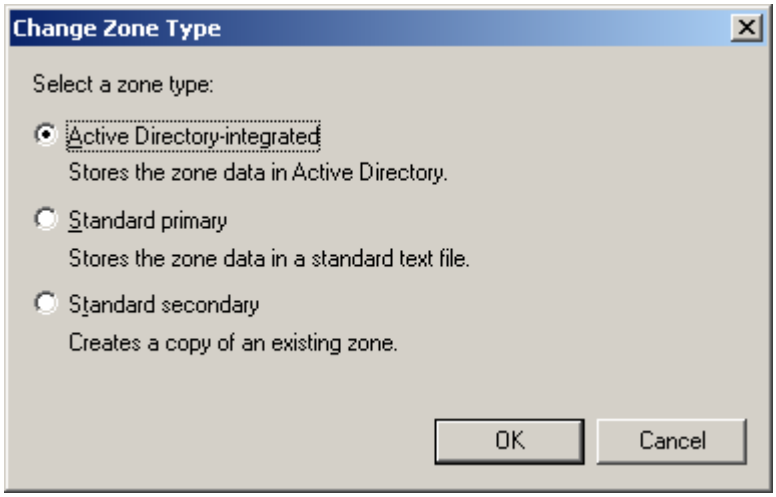
The requirement is as follows:

- *(DNS0800: CAT II) The IAO will ensure that firewall rules or router ACLs prevent unauthorized hosts from outside the enclave from querying Windows 2000 DNS servers.*

5.3 Secure Dynamic Updates and Active Directory

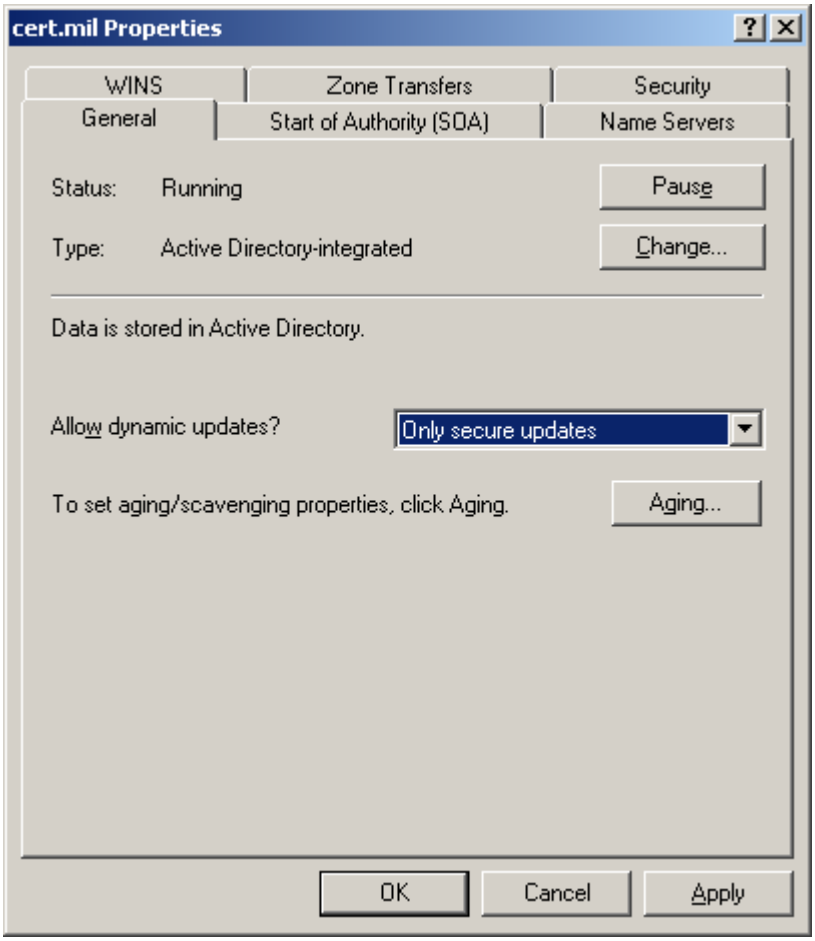
The most compelling reason for using Windows 2000 DNS occurs if there is a requirement for dynamic updates, which must be cryptographically authenticated as specified in *Section 3.4.1.1, Updates of Individual Records*. Windows 2000 DNS provides cryptographic authentication through its Secure Dynamic Updates feature. Appropriate implementation of this capability requires a number of supporting security settings. For example, the feature is only available when DNS is integrated with AD. Moreover, one can only integrate DNS with AD if the DNS service runs on a domain controller.

AD integration can be configured by zone, as is shown in the dialog box below:



Once a zone is configured for AD integration, an administrator can require secure dynamic updates on the "General" tab of the "Properties" dialog box for the zone, as is shown in the following dialog box[9]:

[9] This example and many that follow show a fictitious cert.mil domain. They do not reflect the settings of an actual cert.mil name server.



To prevent unauthorized dynamic updates of records, the ability to modify the records should be limited to the individuals or objects with authority over that record. In many cases, this may be a machine account. Fortunately, SAs can implement these restrictions using AD permissions. A SA would grant a host change permissions to the AD object corresponding to its zone record only and no other objects. This permissions scheme is typically created by default, but SAs should verify that it has been implemented as expected.

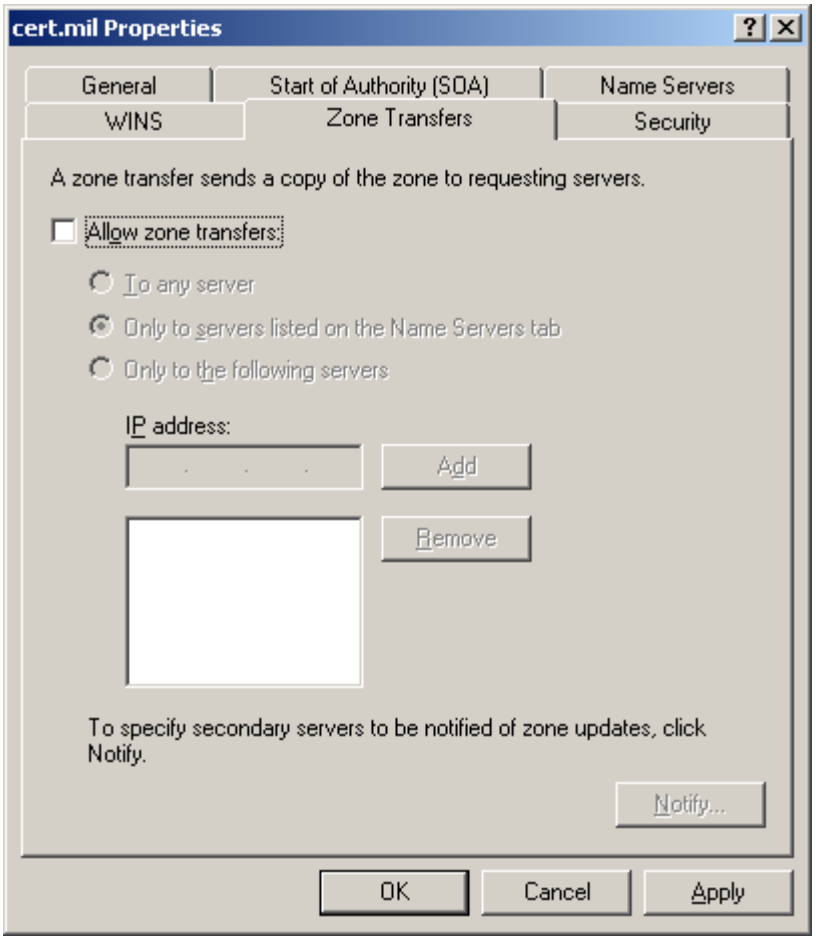
Importantly, AD read permissions usually must be granted to all users because they will need them to get responses to queries. Furthermore, there should be no restrictions on auto-generated SRV records because users typically need to issue queries for these records prior to authenticating. For example, a background process must locate a domain controller to determine which machine the user must communicate with in order to log on. This creates a potential "catch-22." If a user must first log on to query DNS and must query DNS to log on, then the user can accomplish nothing. To resolve this dilemma, the EVERYONE group should be granted read permissions to the LDAP and Kerberos SRV records, which is the default in AD-integrated Windows 2000 DNS. Most SAs are conditioned through STIG guidance and elsewhere to remove the EVERYONE group from all permissions assignments, but this is one case in which that advice should not be followed.

The requirement is as follows:

- *(DNS0805: CAT I) The SA will disable the DHCP server service on any Windows 2000 DNS server that supports dynamic updates.*

5.4 Zone Transfers

Windows 2000 allows for two ways of synchronizing zone data across name servers: (1) traditional RFC-compliant DNS zone transfers; and (2) AD-replication. The latter only works when Windows 2000 DNS is integrated with AD, which makes each of the DNS records an AD object. The Windows 2000 DNS implementation of traditional zone transfers does not meet the STIG requirement that the transfers be cryptographically authenticated using a technology such as TSIG. Fortunately, AD-replication is cryptographically authenticated. Therefore, the solution in a pure Windows 2000 DNS implementation is to integrate DNS with AD and disable zone transfers. This configuration –shown in the example on the following page – will force all zone updates to rely on AD replication exclusively.



In a heterogeneous environment, the problem is more complex. For instance, if both BIND and Windows 2000 DNS name servers are authoritative for the same zone, then they will need to

execute zone transfers, but they cannot employ TSIG because Windows 2000 DNS does not offer this capability.

For this reason, mixed environments are highly discouraged. Instead, DNS software administrators should create subordinate zones for Microsoft services. For example if BIND servers are authoritative for hosts located in kalamazoo.disa.mil, then hosts that require secure dynamic update can be placed in a lower-level domain named ms.kalamazoo.disa.mil or ad.kalamazoo.disa.mil. Windows 2000 DNS servers will be authoritative for the lower-level domain. The name servers authoritative for kalamazoo.disa.mil will reference the lower-level Windows 2000 DNS servers through NS records.

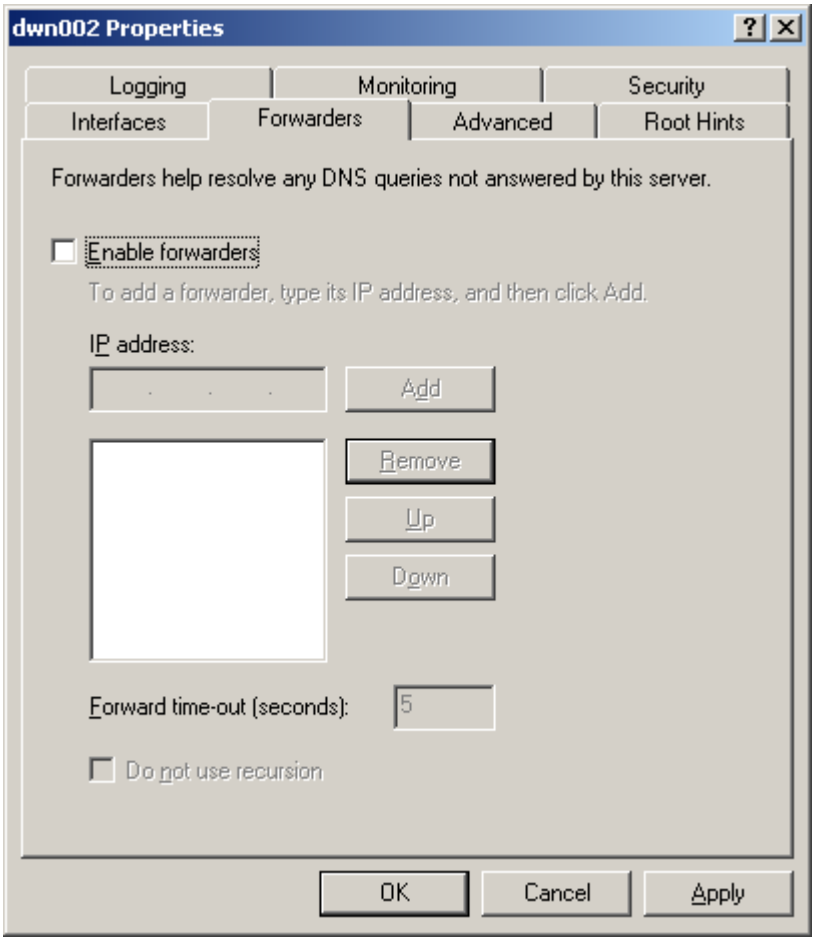
In cases in which domain segregation is not feasible, zone transfers must still be cryptographically authenticated with an alternative to TSIG. The most practical alternative is to create host-to-host virtual private network (VPN) connections between the name servers that need to perform zone transfers. The keys used to encrypt communication between the name servers serve the role that would have been filled by the TSIG keys had both ends of the transaction been able to support TSIG.

The requirement is as follows:

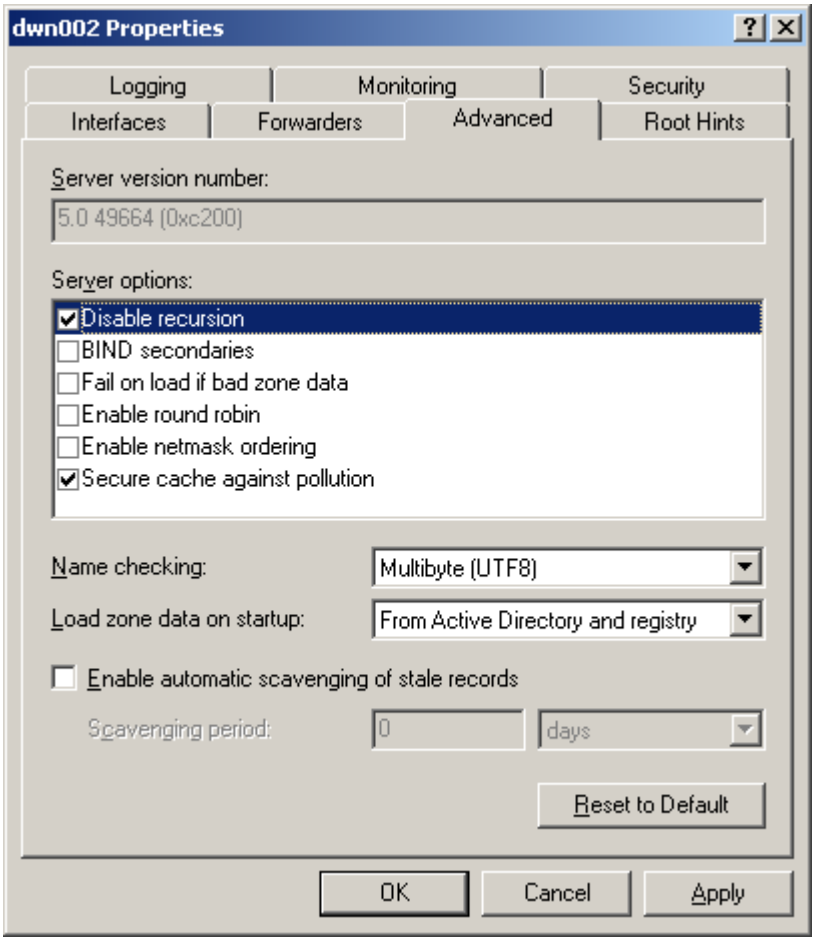
- *(DNS0810: CAT I) The SA will either configure Windows 2000 DNS to prohibit zone transfers or will implement a VPN solution that requires cryptographic authentication of communicating devices and is used exclusively by name servers authoritative for the zone.*

5.5 Forwarders and Recursion

Windows 2000 DNS should not be deployed as a caching name server. Consequently, the use of forwarders and recursion is prohibited on Windows 2000 DNS. To preclude forwarding, an administrator can make sure that the "Enable Forwarders" check box is not marked on the “Forwarders” tab of the name server properties, as shown below:



To prohibit recursion, the administrator can select the server option "Disable recursion" on the “Advanced” tab of the name server properties. The administrator can also select "Secure cache against pollution." Recall that the rationale for prohibiting recursion is to do just that. These settings are shown in the illustration that follows on the next page.



The requirements are as follows:

- *(DNS0815: CAT II) The SA will disable forwarders on an authoritative Windows 2000 DNS server.*

- *(DNS0820: CAT II) The SA will disable recursion on an authoritative Windows 2000 DNS server.*

The configuration shown above is also necessary to comply with the general requirement in *Section 3.4.2.2, Restrictions on Recursion,* that the DNS software administrator prohibit recursion on authoritative name servers.

5.6 WINS Integration

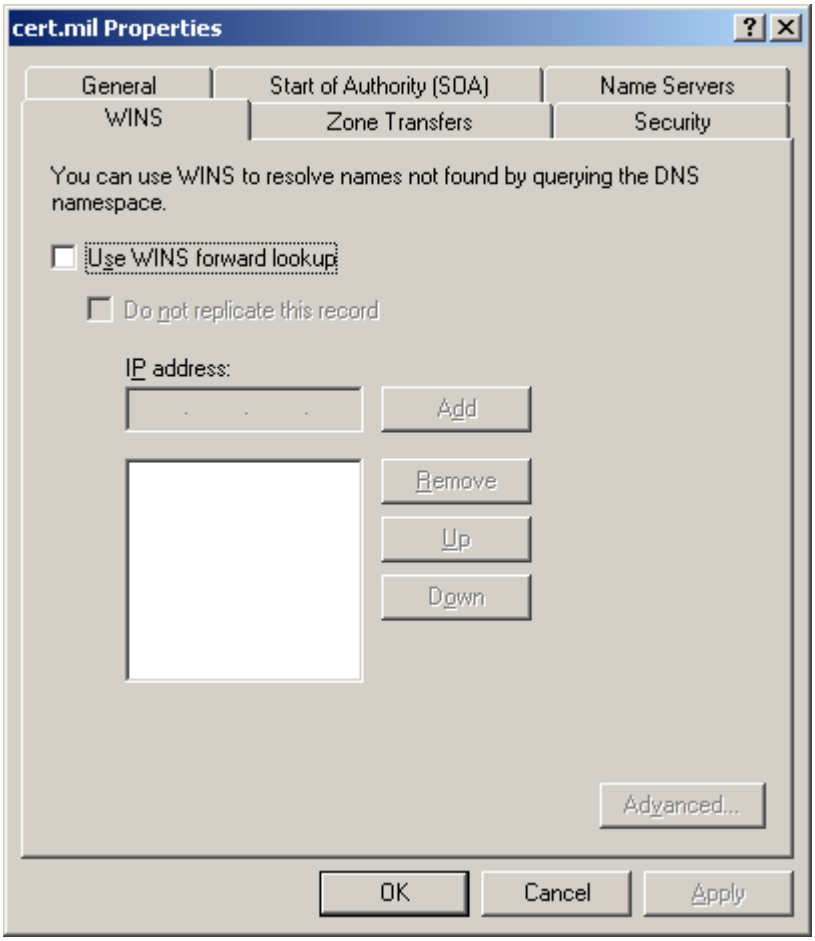
Prior to Windows 2000, Windows operating systems utilized NetBIOS rather than DNS names for most internal network communication, including file, print and messaging services. Microsoft developed the Windows Internet Naming Service (WINS) to provide a central host-name-to-IP-address resolution capability and other services similar to those provided by DNS. WINS, however, supports Windows NetBIOS clients exclusively.

When Microsoft developed its own DNS server software, it included options for DNS/WINS integration. This feature allows a client to first query DNS, which forwards the request to WINS if the record is not found. Importantly, this integration is not required to resolve names because clients can also be configured to send a query to a WINS server when DNS replies that a host name or IP address cannot be resolved.

WINS is insecure because, among other reasons, it will accept dynamic updates without authentication, thereby allowing adversaries to easily modify WINS records. Integrating DNS and WINS creates the potential for DNS to be poisoned with bogus entries in WINS. Therefore, administrators must never integrate WINS with DNS.

With the introduction of Windows 2000, Microsoft embraced DNS as the primary mechanism for host name definition and address resolution. For networks that deploy only Windows 2000 or later versions of Windows, there is no need for WINS. On these networks, DNS can support all desired functionality. WINS is still required for some legacy Windows NT and Windows 9x applications, but pre-2000 Windows computers can be configured to query DNS and WINS separately, eliminating any need for DNS/WINS integration.

To ensure that DNS is not integrated with WINS on a Windows 2000 DNS server, an administrator should make sure the "Use WINS forward lookup" is not checked on the “WINS” tab on the properties dialog of each zone, as shown on the following page.



The requirement is as follows:

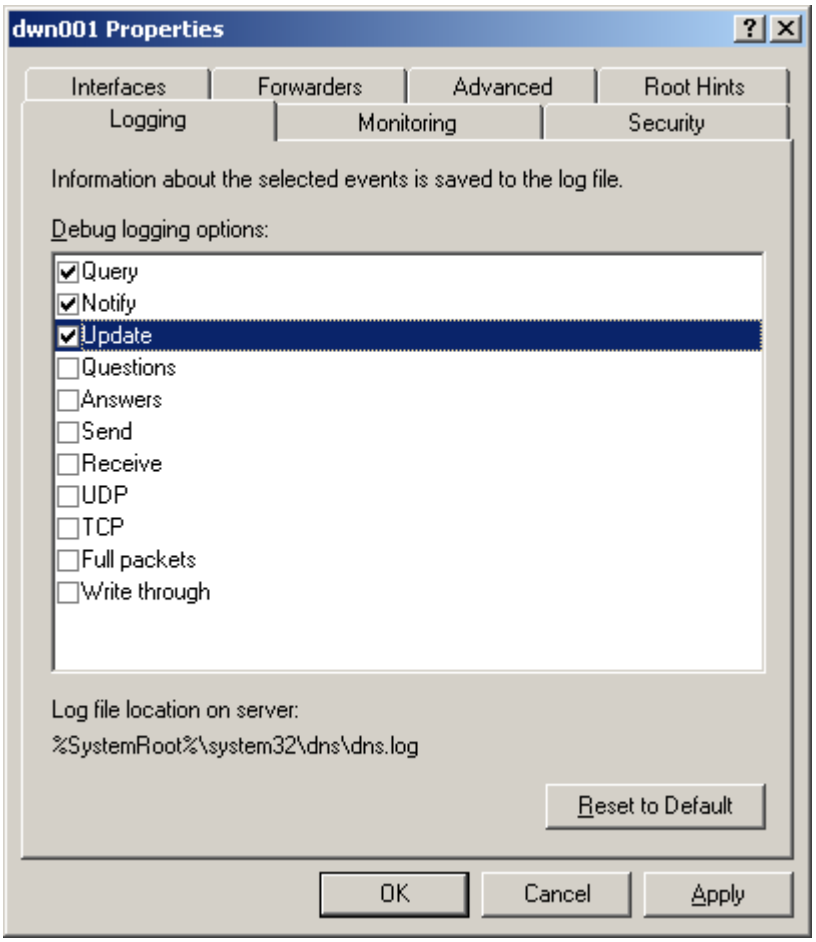
- *(DNS0825: CAT I) The SA will configure Windows 2000 DNS to prohibit WINS lookup.*

5.7 Logging

The general requirements for logging (as specified in *Section 3.5, Logging*) are that the DNS service should log success and failure of the following:

- Start and stop of the DNS service
- Zone transfers
- Zone update notifications
- Dynamic updates
- Queries

Events related to the start and stop of the DNS service appear in the Windows 2000 System Event Log. Other events are logged to a file named *%systemroot%\system32\dns\dns.log*. Windows 2000 DNS has its own logging facility, which is primarily designed to debug DNS problems rather than maintain a record of DNS transactions. To implement the general logging requirements specified in *Section 3.5, Logging*, the DNS software administrator must select the "Query," "Notify," and "Update" debug logging options as shown in the dialog box below.



This page is intentionally left blank.

6. CONFIGURATION OF CISCO CSS DNS

This section provides guidance for the configuration of Cisco Content Services Switch (CSS) DNS. *Section 6.1, CSS DNS Background,* provides background on CSS and its implementation of DNS, discussing in particular its advantages and disadvantages relative to the BIND implementation. *Section 6.2, CSS DNS Zone Delegation,* discusses how CSS zones fit into an overall DNS domain architecture. The remaining sections address security requirements for CSS DNS - *Section 6.3, Simulation of Zone Transfer Restrictions in CSS DNS,* addresses the CSS DNS implementation of zone transfers, and *Section 6.4, Recursion and SDD DNS Forwarders,* discusses query restrictions.

6.1 CSS DNS Background

Cisco CSS is a network device that optimizes a user's Internet experience by directing the user to the server best able to respond to the user's request. At a basic level, this can involve load balancing, in which requests are distributed among several servers to optimize the processing load on each. More sophisticated CSS architectures utilize content routing, in which CSS forwards a user to the *closest* server based on the location of the client (i.e., the one with the fastest current response time given that location). CSS DNS can also sense when a server is down and temporarily remove host records for that server until it is back on line.

CSS DNS is a core technology behind content routing. For example, suppose identically configured web servers around the world supported a web site with the URL www.test.mil. CSS DNS might provide an Air Force officer in the Pentagon with the IP address of a www.test.mil server in Quantico, Virginia while CSS DNS might provide a Marine at Pearl Harbor the IP address of a server with identical content in Hawaii. In the absence of content routing, both the warfighter in the Pentagon and at Pearl Harbor would be able to traverse NIPRNet to a server on the other side of the globe, but content routing provides faster response times by directing users to servers closer by. Moreover, it utilizes the network more efficiently because traffic does not have to travel over long-haul routes.

Traditional implementations of DNS, such as BIND, cannot support this functionality. At best, these traditional implementations provide a *round robin* capability that would alternate between providing a user the IP address of the server at Quantico server and the one at Pearl Harbor, but this crude form of load balancing does not optimize the user experience in the same manner as content routing. In addition, the traditional implementations of DNS do modify response behavior when a server is temporarily unavailable. Thus, good reasons exist for an organization that has geographically distributed web services to want to deploy a DNS product with the features present in Cisco CSS DNS.

On the other hand, Cisco CSS DNS lacks many of the key security features of BIND, including TSIG authentication and access control lists. Therefore, without mitigating controls – which will be discussed later in this section – Cisco CSS DNS' default configuration provides considerably less protection against DNS attacks than a BIND name server that complies with the requirements listed in this STIG.

Cisco CSS DNS also lacks support for DNSSEC, which means that DNSSEC-enabled BIND servers will not be able to cryptographically validate any host-to-IP address mappings issued from a CSS name server. Arguably, the value of DNSSEC zone record validation rises with the frequency that a record changes because an adversary that spoofs or changes an IP address is more likely to go unnoticed when the address is already changing often under normal conditions. Indeed, CSS DNS' ability to rapidly change the DNS response to a query based on current conditions is precisely the environment under which a user would most want the assurance that DNSSEC provides. Consequently, at this time, the deployment of CSS DNS (and similar technologies) is a significant impediment for the successful adoption of DNSSEC.

6.2 CSS DNS Zone Delegation

CSS DNS plays a special role in improving the performance of key applications, but in most large enterprises, BIND and Windows name servers dominate the DNS infrastructure. CSS DNS must fit into the enterprise DNS architecture to be a functional component of that architecture. However, CSS DNS does not currently support zone transfers with non-CSS DNS name servers. Accordingly, its interaction with other name servers must involve either hosting child zones of higher-level zones (likely supported by BIND name servers), or serving as a parent to lower-level zones.

6.2.1 CSS DNS Childhood

CSS DNS *childhood* refers to cases in which the CSS DNS authoritative records are in a child zone of a higher-level parent zone. So that readers can better appreciate security concerns and requirements, we will use the www.test.mil example introduced earlier to explain on a basic level how CSS DNS zone delegation works in practice. CSS DNS is highly complex and a complete understanding of its options and behavior is beyond the scope of this STIG.

In a typical DNS implementation (i.e., one not utilizing CSS DNS), www.test.mil would be defined as an A record in a zone file. If www.test.mil were actually several servers supporting a virtual host, there might be several A records. The records would appear in a zone file shown in Partial Zone File Example 1:

Partial Zone File Example 1: Traditional DNS

```
$ORIGIN test.mil
www       IN   A    150.10.34.200       ; Quantico server
          IN   A    254.35.39.5         ; Pearl Harbor server
```

This configuration provides a round robin effect in which the name server alternates between providing the address of the Quantico and Pearl Harbor servers in response to queries for www.test.mil. As discussed, the round robin load balancing behavior is not equivalent to content routing, in which DNS clients would be provided either the Quantico or Pearl Harbor IP address depending upon the location of the client. To implement a content routing architecture using CSS DNS, the zone file supporting the test.mil zone might appear as follows:

Partial Zone File Example 2: Delegation to CSS DNS with NS and Glue Records

```
$ORIGIN test.mil
www       IN   NS   css101.csd.disa.mil.
          IN   NS   css201.csd.disa.mil.
css101.csd.disa.mil.     IN   A    150.10.34.2
css201.csd.disa.mil.     IN   A    254.35.39.2
```

In Example 2, the name servers css101 and css201 are CSS devices in the csd.disa.mil domain. Importantly, this zone file might reside on a BIND name server, whose function might be merely to provide the DNS glue to the content routing CSS devices. Given the configuration in the zone file above, the CSS devices are authoritative for www.test.mil and, therefore, can determine whether to respond to www.test.mil queries with the Quantico or Pearl Harbor addresses based on internal configuration and algorithms.

In the first example, the server hosting the test.mil zone file maintains authority for www.test.mil through its A records in that file, a traditional approach that limits functionality. In the second example, authority for www.test.mil is delegated to CSS name servers, which can use their content routing rules to determine how to best respond to queries for www.test.mil.

The CSS commands to support this configuration on css101 might be as follows (using the specialized CSS DNS configuration syntax):

```
(config)# dns-server zone 0 tier1 “US-East” leastloaded
(config)# dns-record a www.test.mil 150.10.34.200 single kal-
icmp sticky-enabled proximity
```

Similarly, the commands on css201 might be as follows:

```
(config)# dns-server zone 1 tier1 “Pacific” script
(config)# dns-record a www.test.mil 254.35.39.5 single kal-icmp
sticky-enabled script
```

As mentioned, the full scope of CSS command syntax and functionality is beyond the scope of this STIG. Nonetheless, several elements of these commands illustrate some of the features of the CSS DNS implementation. For instance, note that in CSS DNS zone 0, the A record for www.test.mil server resolves to the Quantico IP address, while in zone 1 it resolves to the Pearl Harbor IP address. In addition, the descriptions “US-East” and “Pacific” suggest that the zones correspond to geographic regions, which can assist with content routing decisions. Indeed, the final term in the *dns-record* command controls that behavior. In zone 0, responses to client

queries for www.test.mil are based on a proximity database. In zone 1, the responses are based on the scrip method, which returns zone records based on a source IP address hash. Both methods use an algorithm to return an IP address to the DNS client that should improve the user's experience.

The primary security concern with regard to the type of delegation discussed above is that to implement this approach, an organization would have to migrate its authoritative records from a well-known DNS implementation with proven, tested security controls to a relatively new DNS implementation without similar controls. Therefore, this migration should only occur when the performance and availability advantages of CSS significantly outweigh the increased residual security risk of using a less mature technology.

The requirement is as follows:

- *(DNS0905: CAT II) The IAO will ensure that Cisco CSS DNS is not utilized to host the organization's authoritative records unless DISA Computing Services supports that host in its csd.disa.mil domain and associated high-availability server infrastructure.*

There are no security requirements related to specific CSS DNS configuration options related to method of content routing or content rules in general, but DISA may issue guidance at a future date if security vulnerabilities with a particular method or algorithm arise.

Cisco CSS utilizes load distribution algorithms that have the capability to route client requests based on server availability, network congestion levels, and the relative proximity of the client to each server, while BIND does not have these capabilities. Both BIND and Cisco CSS DNS support round-robin resolution in which the name server rotates through a list of IP addresses associated with a host name. If an organization only needs the round-robin capability and not the more sophisticated load distribution algorithms, then it should implement BIND because it is more mature and has greater security functionality than Cisco CSS DNS.

6.2.2 CSS DNS Parenthood

The previous section reviewed how a traditional DNS implementation (perhaps utilizing BIND) could delegate authority to CSS DNS so that an organization can take advantage of features that CSS DNS offers but BIND and many other DNS implementations do not. In this section, we review the inverse situation in which a CSS DNS server delegates authority from one of its zones to a non-CSS DNS server that supports subordinate zones.

Although it is technically possible to delegate zones within CSS DNS, there is almost never a rationale to do so because such delegation could be achieved as easily with BIND, which offers security features not present in CSS DNS. Moreover, the performance enhancing features of CSS typically would not apply to name server records because these records are obtained easily and quickly across the wide area without significant impact on a user's experience.

To eliminate unnecessary NS records and related zone delegations from CSS DNS, an administrator can use the following command:

```
(config)# no dns-record ns domain_name
```

The requirement is as follows:

- *(DNS0910: CAT III) The Systems Administrator will not delegate zones in Cisco CSS DNS.*

6.3 Simulation of Zone Transfer Restrictions in CSS DNS

CSS DNS does not support RFC-compliant zone transfers. Instead, it shares information related to DNS records with other CSS devices using the Application Peering Protocol (APP). Although APP sessions are substantially different from zone transfers in practice, they still must meet the same general security requirements because their underlying purpose (to share information on zone records) is similar.

The core requirements related to zone transfers are that an authoritative name server transfers zone information only to designated zone partners and that name servers only accept zone data when it is cryptographically authenticated.

CSS APP provides means to designate with which devices it can share zone data and to authenticate those transactions. CSS devices can define their peers using IP addresses and authenticate them using Challenge Handshake Authentication Protocol (CHAP) with a shared-secret. This setup also can be supplemented with MD5 hashing encryption. While this configuration does not provide the equivalent strength of cryptographic authentication as BIND's TSIG HMAC-MD5, it does provide a satisfactory level of information assurance when CSS DNS operates within a trusted network environment.

To activate these features, the CSS Systems Administrator can issue the following command for each designated peer:

```
(config)# app session ip_address authChallenge shared_secret
encryptMd5hash
```

In this command, *ip_address* refers to the IP address of the designated peer and the shared_secret is a text string up to 32 characters in length.

Other security controls also exist outside of CSS to enhance information assurance. For instance as an alternative to using the built-in authentication features of CSS APP, one can place IPSEC VPN routers in front of each CSS device that will serve to both authenticate end points and provide confidentiality to transferred data. One can also limit APP communication to an out-of-band network, which would make it considerably more difficult for adversaries to spoof the addresses of peers or hijack APP sessions.

The requirements are as follows:

- *(DNS0915: CAT I) The SA will activate APP CHAP authentication and MD5 hashing features for APP sessions with each APP peer with which a CSS device communicates. Alternatively, the SA will achieve authentication and encryption using an IPSEC VPN between each peer couple.*

- *(DNS0900: CAT III) The shared secret in any APP session will be a randomly generated 32-character text string.*

- *(DNS0920: CAT III) The SA will configure CSS to transmit APP session data over an out-of-band network if one is available.*

6.4 Simulation of Query Restrictions in CSS DNS

CSS DNS does not support query restrictions, which represents a significant vulnerability in environments in which these restrictions are necessary to assure the confidentiality of zone data. The core DNS query restriction applicable to all DNS implementations is that internal records only be shared with relevant trustworthy clients. This requirement leads to split-DNS implementations in cases in which a DNS zone must support both DMZ hosts that are accessed from outside the enclave and internal hosts that are accessed inside the enclave.

Fortunately, the absence of query restrictions in CSS DNS should not pose a security problem for most CSS applications. CSS technology is focused on optimizing the performance of high-volume, highly-distributed application systems. It is likely that these application systems are either available to the public or a wide range of DOD users located in a number of different geographic locations. In this environment, a split-DNS is inappropriate because CSS DNS does not provide clients with IP addresses of hosts inside a protected enclave – most likely all of its authoritative records should be available to a broad audience.

In cases in which queries must be limited for a valid business reason, the IAO should implement one of the following alternatives to query restrictions within CSS DNS to mitigate the risk of their absence:

- Query restrictions within the relevant zone statement in the DNS configuration file on the name servers supporting the parent zone to the CSS DNS records (which prevents clients from learning the NS records required to reach the CSS devices).

- Router or firewall access control lists that only allow authorized hosts to access the CSS devices or the servers that they support.

- Security modules within the CSS-supported application that authenticate clients through PKI certificates, source IP address, or .mil reverse lookup.

6.5 Recursion and CSS DNS Forwarders

CSS DNS is not vulnerable to attacks associated with recursion because it does not support recursion, but does offer a forwarder feature that sends un-resolvable or unsupported requests to another name server. This feature poses a risk because the forwarder feature merely redirects potential attacks to another name server.

If zone delegation is implemented as described in *Section 6.2.1, CSS DNS Childhood*, no reason exists for CSS DNS to receive an un-resolvable or unsupported query unless the query is indeed un-resolvable or unsupported. Although these queries may occur unintentionally, many sophisticated DNS attacks rely on improper handling of malformed requests to achieve their desired impact, whether that is to crash the server, poison the record cache, or obtain the privileges of the DNS process. In this environment, CSS DNS forwarding may be a back door means of reaching a name server than would have otherwise been inaccessible to the attacker. Given the near certainty that the forwarder will not provide requested records to a legitimate client if a STIG-compliant CSS DNS cannot do so, the best course of action is to drop these requests rather than forward them.

To disable forwarders in CSS DNS, the Systems Administrator enters the following commands:

```
(config)# no dns-server forwarder primary
(config)# no dns-server forwarder secondary
```

The requirement is as follows:

- *(DNS0925: CAT II) The SA will disable forwarders in Cisco CSS DNS.*

This page is intentionally left blank.

7. STANDARD OPERATING PROCEDURES FOR DNS

To secure DNS, one must not only implement a secure baseline configuration, but one must manage that configuration securely from that point forward. One must also monitor DNS transactions and plan for contingencies. This section covers these operational aspects of DNS security.

A recommended reference is RFC 2870 (Root Name Server Operational Requirements). This RFC lists mandates for root servers, but its guidance is of interest to anyone with responsibility for a name server at any level in the DNS hierarchy. The IAO at each facility should review this document and ensure that all DNS administrators at the site do so as well. The *Action Plan for DOD DNS Hardening* requires that administrators of authoritative DNS servers at the second and third level of the DOD hierarchy (i.e., .mil and dod-component.mil) review this document thoroughly.

Standard Operating Procedures impact individuals in a variety of DNS support roles, including security management, systems administration, DNS software administration, and DNS database administration. Each of these roles is addressed in the subsections that follow.

7.1 Security Management Responsibilities

Security management involves the tracking of security-relevant personnel assignments, physical access control, and business continuity. The IAO will either have direct responsibilities for these areas or work closely with those that do.

7.1.1 Personnel

The security of the DOD DNS infrastructure depends upon the availability and integrity of the professionals that support it.

The requirements are as follows:

- *(DNS0120: CAT IV) The IAO will maintain a list of personnel authorized to administer each zone and name server over which he or she has responsibility.*

- *(DNS0125: CAT II) The IAO will ensure that there is at least one backup DNS database administrator identified for each supported zone and at least one backup DNS software administrator for each name server.*

7.1.2 Physical Access Control

RFC 2870 states that the specific area in which the server is located must have positive access control. In other words, the number of individuals permitted access to the area must be limited, controlled, and recorded. At a minimum, control measures should include mechanical or electronic locks.

These protection mechanisms are highly recommended but may not be feasible at all sites. Name servers, however, should be among the most secured computers at a location because compromise of a name server can directly impact the security of the services it supports. For example, a facility should not house web servers in a locked cage within a secured data center, but allow a name server that resolves the IP addresses for those web servers to reside on an administrator's desk.

The requirement is as follows:

- *(DNS0100: CAT II) The IAO will ensure that a caching name server is protected by equivalent or better physical access controls than the clients it supports.*

- *(DNS0105: CAT II) The IAO will ensure that an authoritative name server is protected by equivalent or better physical access controls than each of the hosts in its zone.*

7.1.3 Business Continuity

Business continuity involves planning for contingencies in which an entire site is lost due to a terrorist attack, fire, or severe natural disaster. In general, the strategy must be to move production computing services to another location prepared to handle them. To accomplish this task, hardware, software, and data must be present or easily obtainable at the other location. Any such planning must include DNS if it is to be effective.

In the case of DNS, a well-designed architecture might have already placed an operating name server (e.g., a zone slave) at the disaster recovery site. If cost or other constraints preclude this, the IAO must, at a minimum, ensure that an off-site copy of zone information exists (preferably in a digital format but in a hard copy format otherwise) to prevent complete loss of records in the event of a disaster.

A key component of business continuity is continuity of electrical power. For root servers, RFC requires power continuity of at least 48 hours, which essentially necessitates on-site electric generators. Of course, not all sites have this capability or are able procure it. Nevertheless, if all the name servers supporting a zone lose power, users may be unable to reach the hosts defined in the zone because they cannot resolve the host's name to its IP address. Therefore, name servers should have Uninterruptible Power Supply (UPS) or alternative power source similar to the hosts that they support.

7.2 System Administration Responsibilities

The Systems Administrator of a computer running name server software has the responsibility to ensure that security patches on the computer are current and that critical files are backed up.

7.2.1 Software Patches

DNS software has a history of vulnerabilities and new ones may be discovered at any time. To ensure that attackers cannot take advantage of known DNS vulnerabilities applicable patches and patch documentation must be maintained. Guidance on appropriate patches is included in IAVM

notices issued by the DOD Computer Emergency Response Team (CERT). *Appendix B* contains a list of IAVM notices relevant to DNS.

The requirement is as follows:

- *(DNS0130: CAT II) The SA will maintain a log of the date and time each patch or upgrade to DNS software was implemented.*

7.2.2 Backup

Fortunately, the DNS architecture is such that there should always be a hot backup of zone information present whenever the primary name server is unavailable for any reason (i.e., the authoritative slave server maintains a copy of the zone files on the master). This built-in redundancy, however, does not extend to configuration files and logs. Therefore, name servers should be backed up to an external media (e.g., tape, optical disk, etc.) on a regular basis.

At some locations, an automated enterprise backup system supports many servers. In this case, name servers can simply be added to the enterprise system. At other locations, backups must be performed manually, placing a considerably higher burden on administrators. In circumstances in which zone and configuration information is very static, remaining the same for several months at a time, it would make little sense to conduct daily full backups. Backups should occur as frequently as needed to capture changes on the name server.

The requirement is as follows:

- *(DNS0135: CAT II) The IAO will establish operating procedures that will ensure that, at a minimum, DNS configuration and resource record data is backed up on any day on which there are changes.*

7.3 DNS Software Administration Responsibilities

The DNS software administrator maintains DNS configuration files and DNS-related cryptographic keys, and reviews DNS logs for security-related events.

7.3.1 DNS Configuration File Changes

Much of this document explains how configuration file settings affect system security. Consequently, changes to configuration files are likely to impact security and, therefore, must be tracked.

The requirement is as follows:

- *(DNS0140: CAT II) The IAO will maintain a log of the date and time any DNS configuration file was modified and the business justification for that modification.*

7.3.2 Cryptographic Key Supersession

Like user account passwords, cryptographic keys such as TSIG keys must be changed periodically to minimize the probability that they will be compromised. If there is a known compromise of a TSIG key, then it needs to be replaced immediately. One of the most important aspects of *key supersession* is the method that will be used to transfer newly generated keys. Possibilities, in rough order of preference, are as follows:

- SSH
- Encrypted e-mail using DOD PKI certificates (preferred) or PGP
- Secure fax (STU-III)
- Regular mail (using the expedited mailing service holding the current GSA contract for "small package overnight delivery service")
- Hand courier

The operational details of key supersession are beyond the scope of this STIG.

The requirements are as follows:

- *(DNS0445: CAT II) The IAO will ensure that a cryptographic key used to secure DNS transactions will never be utilized on a name server for more than one year.*

- *(DNS0145: CAT II) The IAO will establish written procedures for the replacement of cryptographic keys used to secure DNS transactions that will cover, at a minimum:*

  - *Frequency of key supersession*
  - *Criteria for triggering emergency supersession events*
  - *Notification of relevant personnel during emergency and non-emergency supersession*
  - *Methods for securely transferring newly generated keys*

7.3.3 Log Review

This STIG specifies configuration requirements for logging, but if administrators never review the logs that the configuration generates, they are of little value. Enterprises whose personnel resources or DNS transaction logs are such that frequent log review is not feasible should procure log management tools that assist administrators with identifying the most important log entries.

The requirements are as follows:

- *(DNS0110: CAT II) The IAO will ensure that DNS log archival requirements meet or exceed the log archival requirements of the operating system on which the DNS software resides.*

- *(DNS0115: CAT II) The DNS software administrator will review DNS logs daily or employ a real-time log analysis or network management tool that immediately alerts an administrator of critical DNS system messages.*

7.4 DNS Database Administration Responsibilities

One of the simplest changes to DNS may also be one of its greatest potential vulnerabilities, namely adding a host name and its associated IP address to the zone file. Without a rigorous process for adding and modifying resource records, an attacker can social engineer the system (i.e., manipulate human processes rather than circumvent technical controls). For example, an attacker might be able to simply call the DNS database administrator on the phone and successfully request that the IP address of an e-mail server be changed to a rogue server under the attacker's control. Resource records can be modified to compromise security in countless ways.

To best assure the integrity of zone files, one must not only carefully manage the manner in which requests are processed but also periodically check that the current records are valid. For example, when equipment is retired, people often fail to remove the associated host from the DNS. Without periodic checks, an attacker may use a retired host IP address to obtain valuable information from another user who was unaware of the change.

The details of the procedures to add or modify resource records are beyond the scope of this STIG. The requirement is that there be a written procedure in place that meets certain basic criteria.

The requirement is as follows:

- *(DNS0150: CAT II) The IAO will establish written procedures regarding the following:*

  - *The process for updating zone records*
  - *Who is authorized to submit and approve update requests*
  - *How the DNS database administrator verifies the identity of the person from whom he or she received the request*
  - *How the DNS database administrator documents any changes made*

8. DNSSEC

The DNS Security protocol (DNSSEC) is a promising technology for authentication of DNS transactions.[10] It relies on public key cryptography to sign transactions, which, if implemented properly, greatly reduces the administrative burden associated with managing and transferring keys.

A powerful feature of DNSSEC is the ability to sign records to ensure their integrity and authenticity throughout the DNS infrastructure and not just between the authoritative name server and its zone partner or local client. The advantages of this feature become apparent when DOD users wish to securely validate records from other organizations, including commercial vendors, business partners, and other Government agencies.

For example, suppose a user wants assurance that information obtained from what it believes to be cert.mil severs are, in fact, actually cert.mil servers and not an adversary's servers masquerading as cert.mil servers. Presumably, the user or its local caching server could share a TSIG key with the name servers authoritative for the cert.mil zone, but this would be a very cumbersome approach. The cert.mil administrator would have to generate and distribute keys to a wide variety of users. Those users, in turn, would probably want keys for numerous zones, not just cert.mil. For a complete implementation, caching servers would also need TSIG keys for root servers and each top-level domain. Once more than a handful of hosts and domains are involved, the key sharing agreements become so complex that they become impractical. Indeed, this problem is inherent in any symmetric key cryptosystem, of which TSIG is only one instance.

With DNSSEC and its accompanying public key infrastructure (PKI), organizations only need to trust a common signing authority. In the above example, if cert.mil has its zone signed by an authority that the osd.mil name server also trusts, then the desired assurance is achieved. The objective is to establish signing authorities broadly recognized throughout DOD and elsewhere. Although creating the DNSSEC key infrastructure is a challenge, it is much more feasible than acquiring equivalent capabilities with TSIG.

There are no requirements for DNSSEC at this time because the technology is not mature enough to prevent or delay Certification and Accreditation (C&A) of systems that have not yet implemented it. Once DNSSEC has been formally tested in the environment, personnel have been trained, an appropriate public key infrastructure is available, and operating procedures are in place, DNSSEC is likely to be the preferred technology for securing DNS. Organizations can implement DNSSEC today if they choose to do so, but are advised to perform the necessary planning, training and infrastructure upgrades prior to such an implementation.

[10] DNSSEC is sometimes considered to include both Transaction Authentication and signed zones. In this document, DNSSEC refers only to signed zones.

This page is intentionally left blank.

APPENDIX A. RELATED PUBLICATIONS

Publications:

Comer, Douglas. *Internetworking with TCP/IP*. Prentice Hall, 4th Edition, 2000.

Albitz, Paul and Liu, Cricket. *DNS and BIND*. O'Reilly & Associates, Inc., 4th Edition, 2001.

DISA API. Department of Defense Implementation Guidelines for Domain Name System (DNS) Security (DRAFT), 7 September 2000.

DISA Field Security Operations. UNIX Security Technical Implementation Guide, Version 4, Release 4, 15 September 2003.

DISA Field Security Operations. Network Infrastructure Security Technical Implementation Guide, Version 4, Release 2, 15 October 2002.

DISA API. Action Plan for Department of Defense (DOD) Domain Name System (DNS) Hardening (DRAFT), 16 August 2002.

DISA API. Discussion of Advanced Domain Name System (DNS) Features, Windows 2000 Active Directory, and Their Impact Within DOD (DRAFT), 26 August 2002.

ISC, BIND 9 Administration Reference Manual, 2001.

Request For Comment (RFC):

RFCs are currently managed by the Internet Engineering Task Force (IETF) and can be found at http://www.rfc-editor.org.

Recommended RFCs:

RFC 1034, Domain Names – Concepts and Facilities, November 1987
RFC 1035, Domain Names – Implementation and Specification, November 1987
RFC 2181, Clarifications to the DNS Specification, July 1997
RFC 2182, Selection and Operation of Secondary DNS Servers, July 1997
RFC 2535, Domain Name System Security Extensions, March 1999
RFC 2870, Root Name Server Operational Requirements, June 2000

Applicable RFCs:

1713 Tools for DNS Debugging
1912 Common DNS Operational and Configuration Errors
2536 DSA KEYs and SIGs in the Domain Name System (DNS)
2537 RSA/MD5 KEYs and SIGs in the Domain Name System (DNS)
2538 Storing Certificates in the Domain Name System (DNS)
2539 Storage of Diffie-Hellman Keys in the Domain Name System (DNS)
2540 Detached Domain Name System Information
2541 DNS Security Operational Considerations
2845 Secret Key Transaction Authentication for DNS (TSIG)
2930 Secret Key Establishment for DNS (TKEY RR)
2931 DNS Request and Transaction Signatures (TSIG)
3007 Secure Domain Name System (DNS) Dynamic Update
3008 Domain Name System Security Signing Authority

Web Sites:

http://www.isc.org/
Internet Software Consortium

http://www.nominum.com/
Nominum, a for-profit company, is now performing the principal maintenance and development of BIND on behalf of the ISC. Hosts the BIND 9 Administration Reference Manual. (http://www.nominum.com/content/documents/bind9arm.pdf)

http://www.dns.net/dnsrd/
The DNS Resource Directory

http://www.nwfusion.com/research/dns.html
Network World Fusion DNS Resources

http://www.cist-east.saic.com/
SAIC - Center for Information Security. This site is specifically maintained in support of the DOD DNS Security effort.

APPENDIX B. INFORMATION ASSURANCE VULNERABILITY MANAGEMENT (IAVM)

The IAVM notices listed below relate to DNS. Administrators should read these notices and comply with the guidance therein.

2000-B-0001: BIND NXT Buffer Overflow
ftp://www.cert.mil/pub/bulletins/dodcert2000/2000-b-0001.htm
2000-B-0008: BIND 8.2.2-P6 Denial of Service Vulnerabilities
ftp://www.cert.mil/pub/bulletins/dodcert2000/2000-b-0008.htm
2001-A-0001: Multiple Vulnerabilities in BIND
ftp://www.cert.mil/pub/bulletins/dodcert2001/2001-a-0001.htm
2002-A-0006: Multiple Vulnerabilities in ISC BIND 4 and 8
ftp://www.cert.mil/pub/bulletins/dodcert2002/2002-a-0006.htm
2002-T-0010: Denial of Service Vulnerability in ISC-BIND 9
ftp://www.cert.mil/pub/bulletins/dodcert2002/2002-t-0010.htm

This page is intentionally left blank.

APPENDIX C. GENERATION AND STORAGE OF TSIG KEYS

TSIG Keys are required to authenticate zone transfers and updates. This appendix provides instructions on how to create and install a key in a manner that meets the requirements of this STIG. The steps are as follows:

1. Generate the key

From a command prompt on one of the name servers that will utilize the key, change to a directory where the user would like the keys to be created. Ideally, the keys should be located in a designated area of the file system for that purpose (e.g., */etc/dns/keys*/). The command and its results will look similar to the following:

```
% dnssec-keygen -a HMAC-MD5 -b 128 -n HOST example_key.
  Kexample_key.+157+38675
```

Use the HMAC-MD5 algorithm only when the SHA-1 algorithm is not available. BIND 9.2.1 only supports MD5 but future releases should support SHA-1.

This will create two new files in the current directory:

Kexample_key.+157+38675.key
Kexample_key.+157+38675.private

Note that the string that is returned by **dnssec-keygen** is the base name for the two files. The first is followed by ".key" and the second by ".private." The string itself is the name chosen for the key preceded by a "K" and followed by a numerical string related to the key.

An example of the text in the *.key* file is:

```
example_key. IN KEY 512 3 157 2njlQNnzn6HTwKLcjStUXg==
```

An example of the text in the *.private* file is:

```
Private-key-format: v1.2
Algorithm: 157 (HMAC_MD5)
Key: 2njlQNnzn6HTwKLcjStUXg==
```

In this example, the generated key is:

```
2njlQNnzn6HTwKLcjStUXg==
```

Since both files contain the secret key, they should both be protected accordingly. Anyone who obtains one of these files could use the key to perform unauthorized changes to DNS zone records.

2. Create a new designated file for that key

Using a text editor, create a file with the following content:

```
secret "generated_key";
```

In our example, the contents would be:

```
secret "2njlQNnzn6HTwKLcjStUXg==";
```

The syntax of the statement is critical. Ensure that:

- The word "secret" appears at the beginning of the line followed by a space
- The key is included in quotes with no extra spaces before or after the key
- A semi-colon (;) follows the quotation mark after the key
- There are no extra characters, lines, or carriage returns before or after the statement

Importantly, any key longer than approximately 320 bits will contain a space within the key field of the original .key file. This space can be left within the string, as long as it is enclosed within double quotes (") in the new file created to house the key.

3. Transfer the key to the other name server that will utilize the key

Two name servers will need to use the same key to authenticate transactions between them. Therefore, the generated key must be transferred to the other system. If this transfer occurs over a network, then it must be encrypted so that no other party can view the key. If any paper copy of the key is created, then the paper must be shredded after the key is entered on the other system.

4. Set appropriate permissions on key files

As specified in *Appendix E.3.2, TSIG Keys*, TSIG key files should have permissions of 400 and be owned by the user account used to run the name server software.

5. Define the key within the *named.conf* file on each relevant name server

For a key file named *tsig-example.key* that resides in */etc/dns/keys/*, an appropriate *key* statement would be:

```
key example_key {
     algorithm hmac-md5;
     include "/etc/dns/keys/tsig-example.key";
};
```

When *named* is started, it will interpret the include statement and build the following key definition (assuming the generated key in our example):

```
key example_key {
     algorithm hmac-md5;
     secret "2njlQNnzn6HTwKLcjStUXg==";
};
```

This page is intentionally left blank.

APPENDIX D. MODEL STANDARD OPERATING PROCEDURES FOR DNS DATABASE ADMINISTRATION

The model procedures are not mandatory and designed to provide organizations with suggested language for local written procedures. Nevertheless, adoption of the model procedures will ensure that the organization is fully compliant with the administrative requirements related to resource record updates found in *Section 7, Standard Operating Procedures for DNS,* of this STIG. In review, the requirement is that the IAO will establish written procedures regarding the following:

- The process for updating zone records
- Who is authorized to submit and approve update requests
- How the DNS database administrator verifies the identity of the person from whom he or she received the request
- How the DNS database administrator documents any changes made

A key objective of the procedures is separation of duties, which limits the potential for a single insider to misuse the system. There are three key components of any changes to a DNS zone:

- Request
- Approval
- Implementation

Ideally, a different person should perform each of these activities. For example, an application team deploying a new database server would request a host name and IP address for that server. A network security officer or operations manager would approve that request. Finally, a DNS database administrator would implement the approved request.

In some environments, two of the three components may be combined. For example, designated personnel may be authorized to submit requests that require no further approval. In smaller environments, technical staff may perform multiple roles (e.g., IAO and DNS database administrator), in which case the approval and implementation steps may be combined in practice. Combining all three components is ill advised in all but small LAN environments in which one or two IT support professionals perform all IT functions, which makes separation of duties impossible or impractical.

The model procedures are listed below. Optional items in the procedures are placed in brackets ([...]). Comments or guidance within the model procedures are in italics.

1. The authority to request changes to a zone database (e.g., create, modify or delete a record) is limited to (select one of the options below*)*
   [an IAO-maintained list of individuals stored at _____ location]
   [the following organizational roles: _____]

2. The authority to approve change requests is limited to *(*select one of the options below)
   [the IAO]
   [an IAO-maintained list of individuals stored at _____ location]
   [the owner of the domain as specified in the zone database]
   [the DNS database administrator]

3. The authority to implement approved change requests is limited to an IAO-maintained list of DNS database administrators stored at _____ location.

4. The approved method(s) for submitting a requests is/are as follows: *(*select one or more of the options below)
   [send a [digitally signed] e-mail message to the designated approver]
   [send a signed written request to the designated approver]
   [open a ticket with the help desk]

5. Before implementing approved change requests, the DNS database administrator verifies the identity of the individual from whom he or she receives the request: (select one or more of the options below):
   [through voice or face recognition, when the requestor or approver is well known]
   [using the "sent from" field when requests are delivered via e-mail]
   [validating the digital signature in the e-mail client software]
   [through call-back to the requestors desk telephone number]
   [through a comparison of the approving signature on the request to one on file for that individual].

6. The DNS database administrator saves approved requests to _____ location *(*e.g., network file folder, e-mail archive, file cabinet, etc.)

7. When implementing approved change requests, the DNS database administrator documents the following information related the request:
   - the name[, title][, and organizational affiliation] of the requestor
   - the name[, title][, and organizational affiliation] of the approver
   - the name of the DNS database administration implementing the change
   - the date of the change
   [- who is permitted to make subsequent changes to the affected records]

8. Records of updates and changes are located in (select one of the options below)
   [zone files]
   [a database named _____ located at _____]

This page is intentionally left blank.

APPENDIX E. UNIX OPERATING SYSTEM CONFIGURATION TO SUPPORT BIND

The UNIX STIG provides guidance on how to secure a UNIX computer so that it may operate at an acceptable level of risk. That STIG also includes guidance specific to BIND, particularly with respect to file permissions on key BIND files and patches and upgrades needed to comply with DNS-related IAVM notices. This appendix provides additional guidance on OS-level concerns related to BIND. Eventually, the issues will be addressed in the UNIX STIG and, when that occurs, this appendix will be removed.

For general operating system requirements and guidance, refer to *Section 3.2, Name Server Operating System Platforms*.

E.1 Network Services

Only network services absolutely required to ensure the availability, confidentiality, and integrity of DNS information must run on a name server in a DOD DNS infrastructure. Running unnecessary services simply leaves BIND vulnerable to potential exploits related to software not critical to DNS. Therefore, a maximum-security posture prohibits these services.

Listing exactly what services should be available in this STIG is not practical because different sites deploy different hardware, intrusion detection agents, network management tools, and other items required to maintain operations. Nevertheless, the IAO must adhere to the spirit of service minimization to the extent feasible. Clearly, BIND should not run on a server that also hosts a web application, handles e-mail, routes network traffic, or houses a non-DNS database. Both */etc/services* and */etc/inetd.conf* should be configured to reflect this.

Some Systems Administrators like to install BIND on application servers to quicken lookups. For example, an e-mail server that forwards messages to e-mail servers in other domains must issue queries for the MX and A records associated with those other mail servers. If these could be obtained from a locally installed name server, the presence of that name server would speed up query response and reduce network traffic. Nevertheless, the security risks of this configuration outweigh the advantages. In the case of e-mail, the few extra milliseconds it takes to query remote name servers will have no appreciable impact on end-users' e-mail experience, while exposing both e-mail and DNS to each other's security vulnerabilities could have a significant impact on an organizations computing infrastructure.

Beyond *named* and *sshd*, there are a few core services still needed. For example, name servers must comply with all applicable host intrusion detection and system integrity requirements. Consequently, it is likely that products such as Tripwire, Symantec Intruder Alert (ITA), or Symantec Enterprise Security Manager (ESM) are resident on the system. In addition, there may be agent software to support network monitoring or management. If an organization believes it can best ensure 24x7 DNS availability through integration of the name server into site-wide monitoring services, then it is appropriate that software run on the name server to support that service. Similar cases can be made for backup and uninterruptible power supply (UPS) if the site supports these services across the infrastructure with agent software.

The only permitted services to be running on a DNS UNIX BIND server are those implementing:

- DNS
- Secure shell
- Host intrusion detection
- Host file integrity
- Network management or monitoring
- Anti-virus
- Backup
- UPS

The requirement is as follows:

- *(DNS4450: CAT II) At a minimum, the SA will disable the following services in inetd.conf: admind, chargen, echo, etherstatd, fingerd, ftpd, httpd, ICQ server, identd, netstat, netstatd, nit, nntp, nsed, nsemntd, pfilt, portd, quaked, rexd, rexecd, rje_mapper, rlogind, rpc_3270, rpc_alias, rpc_database, rpc_keyserv, rpc_sched, rquotad, rsh, rstatd, rusersd, selectd, serverd, showfhd, sprayd, statmon, sunlink_mapper, sysstat, talkd, telnetd, tfsd, tftpd, timed, ttdb, ugidd, uucpd, and walld.*

- *At a minimum, the SA will disable the following services: NFS client, NFS server process and SNMP daemon, automounter, printer queue daemon, and RPC portmapper. (For Solaris, disable the following scripts in rc2.d: S73nfs.client, S74autofs, S80lp, S71rpc, and S99dtlogin and the following scripts in rc3.d: S15nfs.server and S76snmpd.)*

E.2 Non-privileged User for BIND

The BIND *named* daemon runs as root by default, but this practice represents a security vulnerability because an adversary that managed to gain control of *named* could then control the entire system. To prevent this scenario, BIND can run as a non-privileged user with a UID other than 0 (which belongs to root). The SA, however, must create such an account to implement this approach. For the remainder of this document, it will be assumed that the name of the user is dnsuser and that it is the sole member of a group named dnsgroup.

Unfortunately, the existence of a new non-privileged user introduces a point of entry for a potential attacker. If the attacker can log on as dnsuser, then the attacker would have full control of DNS services. To provide assurance that dnsuser can only be used to run BIND, the SA can, at a minimum, define its shell as */bin/false* (i.e., no shell at all). This prevents an adversary from using the non-privileged user account to execute commands.

The requirements are as follows:

- *(DNS4440: CAT III) The SA will create a special non-privileged DNS user account and configure BIND to run as that user account.*

- *(DNS4460: CAT III) The SA will ensure that no user can obtain a command shell using the DNS user account (e.g., by setting its shell to /bin/false in /etc/passwd or a similar alternative).*

E.3 File Permissions

OS STIGs provide guidance on how SAs can secure a file system, all of which is applicable to name servers. In addition, there are additional files specific to BIND that the SA must also lock down. These include BIND support files and TSIG keys.

E.3.1 BIND Support Files

BIND support files include the configuration files, zone databases, and logs. There is no reason why anyone other than the *named* process and the DNS or System Administrator needs access to these files. The *named* process will need to write to log and zone files, but only the DNS software administrator needs the ability to modify configuration files.

The requirements are as follows:

- *(DNS4470: CAT II) The SA will set ownership and permissions on all BIND name servers at least as restrictive as listed in the table below:*

FILE NAME	*OWNER*	*GROUP*	*PERMISSIONS*
named.conf	*root*	*dnsgroup*	*640*
named.pid	*root*	*dnsgroup*	*600*
Root hints file	*root*	*dnsgroup*	*640*
master zone file	*root*	*dnsgroup*	*640*
slave zone file	*root*	*dnsgroup*	*660*

Common names for the root hints file are *root.hints*, *named.cache*, or *db.cache*. The name is configurable within the *named.conf* file.

- *(DNS4470: CAT III) On BIND 8 name servers, the SA also will set ownership and permissions on all BIND name servers at least as restrictive as listed in the table below (if the listed files exist):*

FILE NAME	*OWNER*	*GROUP*	*PERMISSIONS*
named.run	*root*	*dnsgroup*	*660*
named_dump.db	*root*	*dnsgroup*	*660*
ndc (FIFO)	*root*	*dnsgroup*	*600*
ndc.d (directory containing ndc)	*root*	*dnsgroup*	*700*

Note that dnsgroup (or an alternative name) must be created before its use.

E.3.2 TSIG Keys

Required access to TSIG keys is extremely limited – nothing more than allowing the *named* process the ability to read the keys to sign transactions. Even the DNS and System Administrators should not be able to read or modify the keys. The exception is supersession events – when the keys are replaced. In these cases, the SA should temporarily change file ownership to root and assign write permissions for the duration of the switch only.

FILE NAME	*OWNER*	*GROUP*	*PERMISSIONS*
unique to each key	*dnsuser*	*dnsgroup*	*400*

For detailed requirements and guidance for encryption key file permissions, refer to *Section 3.2, Name Server Operating System Platforms.*

E.3.3 Log Files

Log files should be appropriately secured because they could possibly reveal information about DNS configuration that potentially could assist an adversary attempting to breach the security of DNS. Nevertheless, allowing authenticated users to read the files is still acceptable. If a name server is configured to write DNS messages to the system logging facility, then permissions on these files should follow the relevant OS STIG guidance for this access. If BIND is configured to log events to other files, these logs should be appropriately secured, having file permissions that restrict unauthorized changes.

FILE NAME	*OWNER*	*GROUP*	*PERMISSIONS*
any log file	*dnsuser*	*dnsgroup*	*664*

For detailed requirements guidance for log file permissions, refer to *Section 3.5, Logging.*

E.4 Running BIND in chroot(ed) Directory Structure

With any network service, there is the potential that an attacker can exploit a vulnerability within the program that allows the attacker to gain control of the process and even run system commands with that control. One possible defense against this attack is to strictly limit the software to particular quarantined areas of the file system, memory or both. If such a defense were in place, then even if an attacker gained control of the process, the attacker would be unable to reach other commands or files on the system. This approach often is referred to as a *padded cell*, *jail*, or *sandbox.* All of these terms allude to the fact that the software is contained in an area where it cannot harm either itself or others. A more technical term is a *chroot(ed)* directory structure.

It is highly recommended that the System Administrator configure BIND to run in a padded cell or chroot(ed) directory structure. This configuration is not a strict requirement at this time because additional testing may be required to ensure its feasibility at all sites.

This page is intentionally left blank.

APPENDIX F. MICROSOFT WINDOWS OPERATING SYSTEM CONFIGURATION TO SUPPORT BIND

BIND was written to run on UNIX and other operating systems prior to Windows, but the current source code has been modified to also run on Microsoft Windows NT and Windows 2000. UNIX is the preferred platform for BIND because the behavior of its secure configuration with BIND is better known than the one for Windows. Nevertheless, some sites may select Windows because their staff may be much more familiar with Windows than UNIX or because it may be more cost-effective for them to deploy it.

If a site chooses to install BIND on Windows, then the DNS software administrator should still carefully read the appendix on UNIX configuration that precedes this one. Windows configuration settings related to BIND are derived from corresponding UNIX settings. Thus, to fully understand the motivation behind Windows settings and how they may differ from those in UNIX, the DNS software administrator must have a working knowledge of the secure UNIX configuration.

Eventually, the material in this appendix will be transferred to the Windows STIG. When that transition is complete, this appendix will be removed. For general DNS requirements and guidance related to operating systems, refer to *Section 3.2, Name Server Operating System Platforms.*

F.1 Network Services

The general principle behind acceptable services on Windows is the same as that for UNIX. Only network services absolutely required to ensure the availability, confidentiality, and integrity of DNS information must run on a name server in the DOD DNS infrastructure.

The implementation on Windows, however, is substantially different than the implementation on UNIX. Startup settings for UNIX daemons are defined in *inetd.conf*. In Windows, startup settings for server software are defined in the services applet of the Control Panel or Microsoft Management Console. A UNIX SA can prevent a daemon from running at startup by removing the daemon from *inetd.conf*. A Windows SA can prevent a service from running at startup by clicking on either the "manual" or "disable" options under the configuration dialog box for that service. Using the disable option is strongly preferred for unnecessary services because it prevents unauthorized or accidental use of those services at a later time.

In UNIX, secure remote administration is typically achieved with secure shell (SSH) implemented using the *sshd* daemon. Although there are implementations of SSH for Windows, they are not commonly used and usually require separate licenses. Perhaps the closest equivalent to SSH in Windows is Terminal Services, which provides users with a remote graphical user interface (GUI) as if they were on the machine itself, but without encrypted sessions. Use of Terminal Services is not recommended for remote management of name servers because of potential security vulnerabilities associated with this service.

On DOD local area networks (LANs), a DNS software administrator can remotely administer BIND on Windows via use of standard Windows remote administration tools. The use of the **net start** and **net stop** commands can be used to start and stop the *ISC BIND* service. Text editors such as notepad can be utilized to access remote configuration files. With these tools, a DNS software administrator will be able to perform all required tasks. Relative to SSH, the disadvantage of this type of in-band management is that most communication is not encrypted. Nevertheless, the challenge-response method of authentication at least guarantees that passwords are not sent in clear text.

Remote administration of this sort must be limited to the LAN. If there is a requirement to remotely administer a Windows-based BIND name server from a host on an untrusted network or on the external side of an organization's firewall, then third-party network encryption tools must be utilized to ensure the confidentiality and integrity of all transferred data.

Use of the Windows net commands and remote file access requires that the Server service be running on the name server. To make sure that the Server service is not utilized unnecessarily, there must be no file shares beyond the default administrative shares such as C$.

The only permitted services to be running on a Windows ISC BIND DNS server are those implementing:

- DNS *(i.e., the ISC BIND service)*
- Host intrusion detection
- Host file integrity
- Network management or monitoring
- Anti-virus
- Backup
- UPS

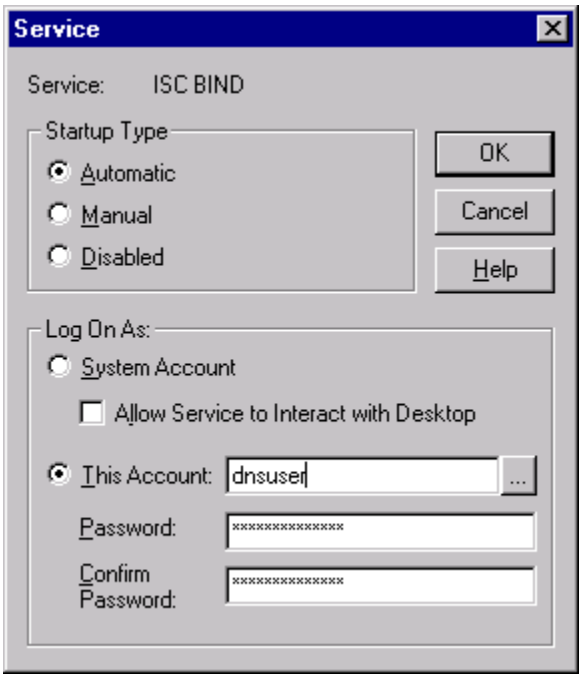
The requirements are as follows:

- *(DNS4570: CAT II) The IAO will ensure that appropriate encryption software is correctly installed and configured on Windows BIND name servers if it is required that in-band remote management be performed from hosts outside the enclave in which the name server resides.*

- *(DNS4580: CAT II) The SA will ensure that no other shares other than the default administrative shares are enabled on a name server.*

F.2 Non-Privileged User for BIND

As in UNIX, the Windows SA can create a special user account dedicated to running BIND. This ensures that if an external entity somehow gains control of BIND, its access to other system resources is restricted to the greatest extent feasible. If this technique were implemented, then the ISC BIND service dialog box would appear as follows:



In Windows, a user does not have to be a member of any group other than the implicit groups "Everyone" and "Authenticated Users." Thus, to best ensure security, dnsuser must be removed from all explicit groups, including the "Users" group, into which all users are placed by default. There should not be a dnsgroup group as is recommended for UNIX.

To log on a service, dnsuser must have the user right "Log on as service." No other rights are necessary. In Windows 2000, users can also be explicitly denied some user rights so that they do not inherit them from group membership. Accordingly, dnsuser must be assigned the user rights "Deny Access to This Computer from the Network" and "Deny Logon as a Batch Job." On properly configured Windows systems dnsuser would never inherit such rights in any case, but the action is a defense-in-depth measure that offers protection when there are authorized changes to the system that could impact dnsuser.

The requirements are as follows:

- *(DNS4530: CAT II) The SA will create an ISC BIND service user account and configure ISC BIND to run as that user.*

- *(DNS4540: CAT IV) The SA will ensure that the ISC BIND service user is not a member of any group whose membership can be configured.*

- *(DNS4550: CAT III) The SA will grant the ISC BIND service the user rights of "log on as service," "Deny Access to This Computer from the Network," and "Deny Logon as a Batch Job," which are required for the proper configuration and security of ISC BIND.*

F.3 File Permissions

On UNIX systems, DNS software administrators will sometimes know the root password to perform additional administrative duties based on roles established at each site and will have the ability to switch (or **su**) to root when performing these duties. In Windows, this type of privilege elevation is uncommon. Instead, DNS administrators should be added to a newly created group named dns-admins. When they log on to their regular user accounts, these users will obtain the permissions needed through their membership in the dns-admins group. An alternative is to work through the administrators group, but this would give DNS administrators far more rights and permissions than is necessary to perform their job functions. Consequently, that approach should be avoided.

The Windows SA must ensure that Windows BIND name servers provide comparable file system protections as UNIX BIND name servers.

The requirement is as follows:

- *(DNS4590: CAT II) The Windows SA will set ownership and permissions on all Windows BIND name servers at least as restrictive as listed in the table below (if a user or group is not listed in the table, then it should not have any permissions to the listed file or folder):*

FOLDER/FILE NAME	***OWNER***	***USER/GROUP***	***PERMISSIONS***
%systemroot%\system32\dns\bin	*Administrators*	*Administrators*	*Full control*
		dns-admins	*Read*
		dnsuser	*Read*
%systemroot%\system32\dns\etc	*Administrators*	*Administrators*	*Full control*
		dns-admins	*Change*
		dnsuser	*Change*
named.conf	*Administrators*	*Administrators*	*Full control*
		dns-admins	*Change*
		dnsuser	*Read*
named.pid	*Administrators*	*Administrators*	*Full control*
		dns-admins	*Read*
		dnsuser	*Change*
named.stat	*Administrators*	*Administrators*	*Full control*
		dns-admins	*Read*
		dnsuser	*Change*
root hints file	*Administrators*	*Administrators*	*Full control*
		dns-admins	*Change*
		dnsuser	*Read*
Any zone file	*Administrators*	*Administrators*	*Full control*
		dns-admins	*Change*
		dnsuser	*Change*
Any TSIG key file	*Administrators*	*dnsuser*	*Read*

Note that dns-admins (or an alternative name) must be created before its use. The folders listed are defaults, but the actual location may vary.

Common names for the root hints file are *root.hints, named.cache,* or *db.cache*. The name is configurable within the *named.conf* file.

This page is intentionally left blank.

APPENDIX G. ROOT HINTS

The following table lists the IP address for each of the Internet root servers. DNS software administrators can use this list to validate the entries in their local root hints files. These addresses rarely change.

Root Server	IP Address
A	198.41.0.4
B	128.9.0.107
C	192.33.4.12
D	128.8.10.90
E	192.203.230.10
F	192.5.5.241
G	192.112.36.4
H	128.63.2.53
I	192.36.148.17
J	192.58.128.30
K	193.0.14.129
L	198.32.64.12
M	202.12.27.33

This page is intentionally left blank.

APPENDIX H. LIST OF ACRONYMS

A	Address (a type of DNS resource record)
ACL	Access Control List
API	Application Engineering – Information Assurance
APP	Application Peering Protocol
BIND	Berkeley Internet Name Domain
C&A	Certification and Accreditation
CDE	Common Desktop Environment
CERT	Computer Emergency Response Team
CHAP	Challenge Handshake Authentication Protocol
CINC	Commanders-in-Chief
CNAME	Canonical Name (a type of DNS resource record)
CS	Computing Services (a division of DISA)
CSS	Content Services Switch
DECC	Defense Enterprise Computing Center
DECC-D	Defense Enterprise Computing Center-Detachment
DHCP	Dynamic Host Configuration Protocol
DIG	Domain Information Groper
DISA	Defense Information Systems Agency
DMS	Defense Message System
DMZ	Demilitarized Zone (literal meaning). (In the context of network security, this refers to a perimeter network connected to a firewall that is not part of the external or internal network but is accessible by external hosts.)
DNS	Domain Name System
DNSSEC	Domain Name System Security
DOD	Department of Defense
DSN	Defense Switched Network
ESM	Enterprise Security Manager
FIPS	Federal Information Processing Standard
FSO	Field Security Operations
FTP	File Transfer Protocol
GCCS	Global Command and Control System
GSS	Generic Security Service
GUI	Graphical User Interface
HMAC	Keyed-Hashing for Message Authentication

IA	Information Assurance
IAM	Information Assurance Manager (formerly ISSM)
IAO	Information Assurance Officer (formerly ISSO)
IASE	Information Assurance Support Environment
IAVA	Information Assurance Vulnerability Alert
IAVM	Information Assurance Vulnerability Management
IOS	Internetworking Operating System
IP	Internet Protocol
IPSEC	Internet Protocol Security
ISC	Internet Software Consortium
ISSM	Information Systems Security Manager
ISSO	Information Systems Security Officer
ITA	Intruder Alert
LAN	Local Area Network
LDAP	Limited Directory Access Protocol
MAC	Mission Assurance Category
MD5	“Message Digest” 5 Algorithm
MX	Mail Destination / Mail Forwarder (a type of DNS resource record)
NIPRNet	Non-classified (but Sensitive) Internet Protocol Router Network
NIST	National Institute of Standards and Technology
NS	Name Server (a type of DNS resource record)
NSA	National Security Agency
OS	Operating System
PDI	Potential Discrepancy Item
PDL	Potential Discrepancy List
PID	Process ID
PKI	Public Key Infrastructure
RFC	Request for Comment
RNOSC	Regional Network Operations and Security Center
RNDC	Remote Name Daemon Control
SA	System Administrator
SHA-1	Secure Hash Algorithm 1
SIPRNet	Secret Internet Protocol Router Network
SMC	Systems Management Center
SOA	Start of Authority
SRR	Security Readiness Review
SRRDB	Security Readiness Review Database
SRV	Service (a type of DNS resource record)
SSH	Secure Shell
SSO	Systems Support Office
STIG	Security Technical Implementation Guide

TCP	Transmission Control Protocol
TLD	Top Level Domain
TS	Terminal Services
TSIG	Transaction Authentication
TXT	Text (a type of DNS resource record)
UDP	User Datagram Protocol
UPS	Uninterruptible Power Supply
URL	Uniform Resource Locator
VCTS	Vulnerability Compliance Tracking System
VMS	Vulnerability Management System
VPN	Virtual Private Network

This page is intentionally left blank.