

TLD Registry Considerations for Secure DNS (DNSSEC)

Jim Reid, DNS-MODA

`jim@dns-moda.org`

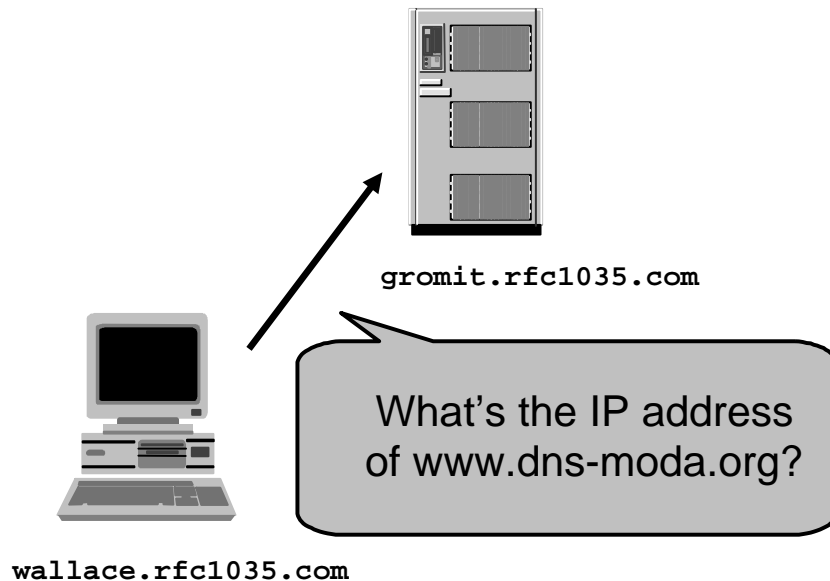
QuickTime™ and a
TIFF (Uncompressed) decompressor
are needed to see this picture.

Introduction

- Why Secure DNS?
 - What DNSSEC can and can't do
- How DNSSEC works (in outline)
- Deployment considerations
- Problems introduced by Secure DNS
- Status of DNSSEC Deployment
- Future work:
 - Protocol development
 - Last-mile issues for deployment & usage

How DNS Lookups Work

- The workstation `wallace.rfc1035.com` queries its local name server, `gromit.rfc1035.com`, for the address of `www.dns-moda.org`



DNS Resolution - 1

- A step-by-step look at the resolution process:

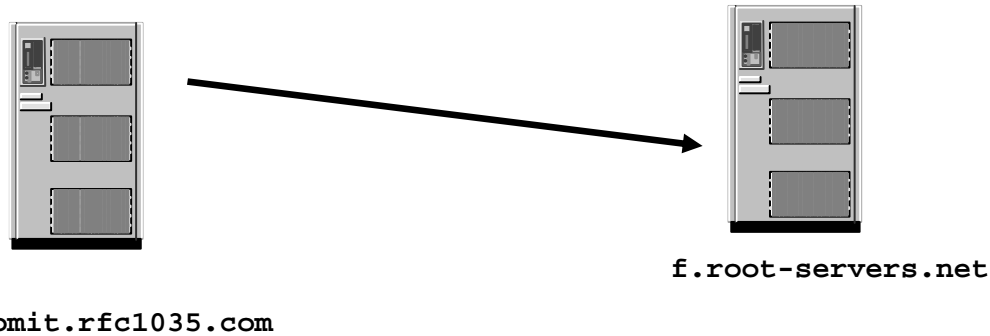


wallace.rfc1035.com

QuickTime™ and a
TIFF (Uncompressed) decompressor
are needed to see this picture.

DNS Resolution - 2

- The name server `gromit` asks a root name server, `f.root-servers.net`, for `www.dns-moda.org`'s address

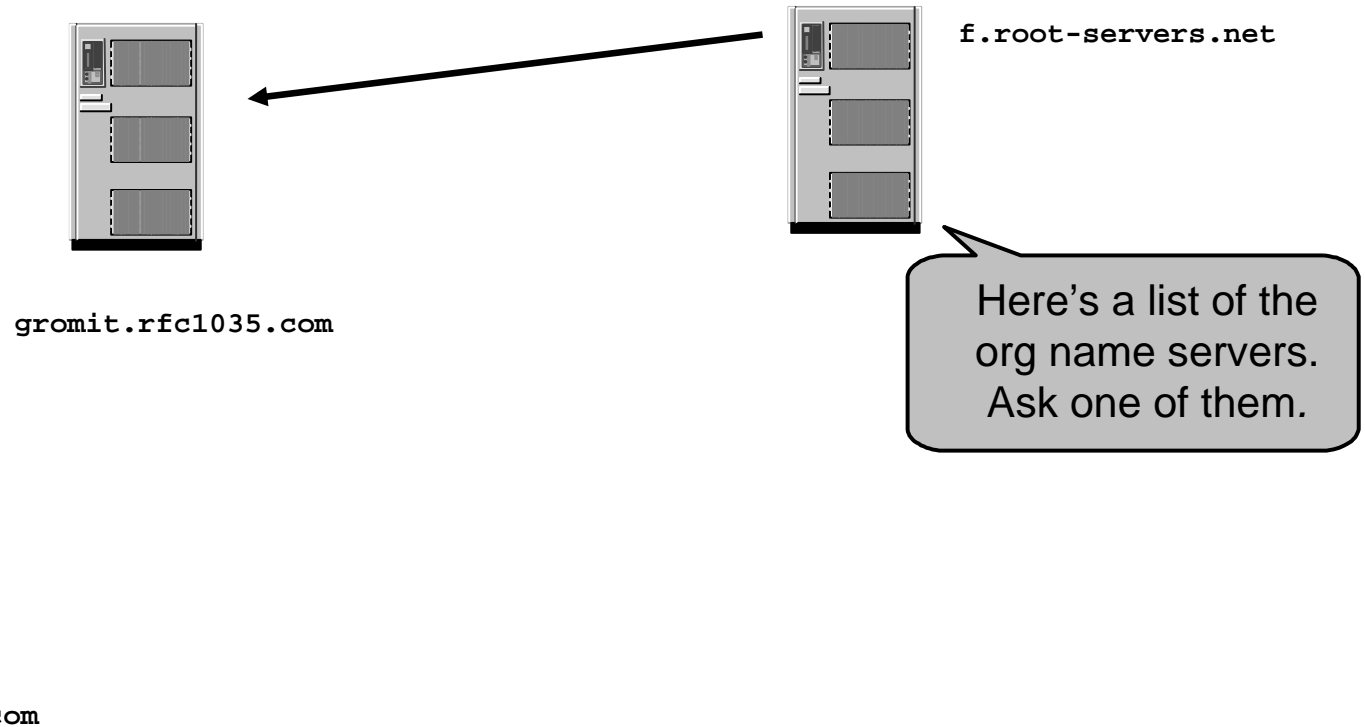


`wallace.rfc1035.com`

What's the IP address of `www.dns-moda.org`?

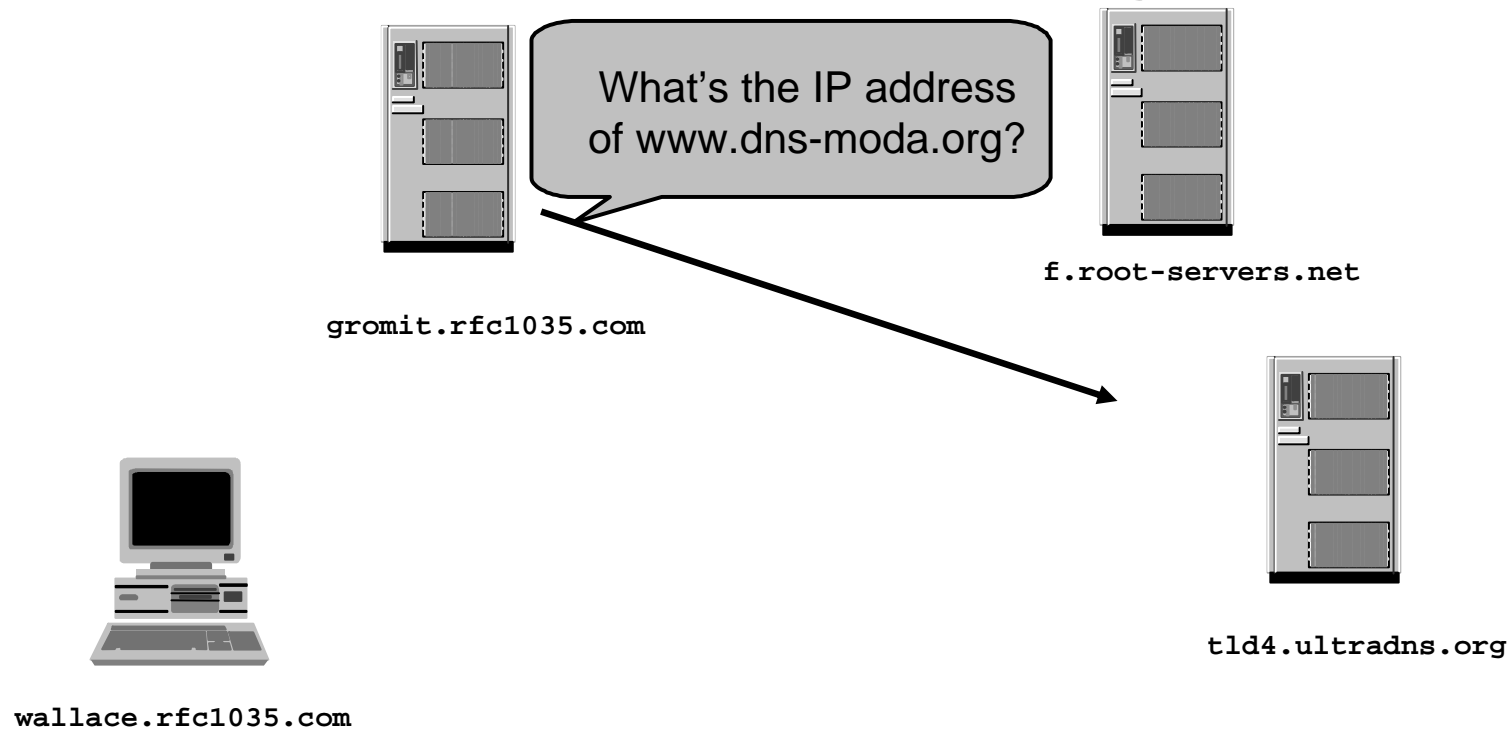
DNS Resolution - 3

- The root server `f` tells `gromit` to query the `org` name servers
- This type of response is known as a *referral*



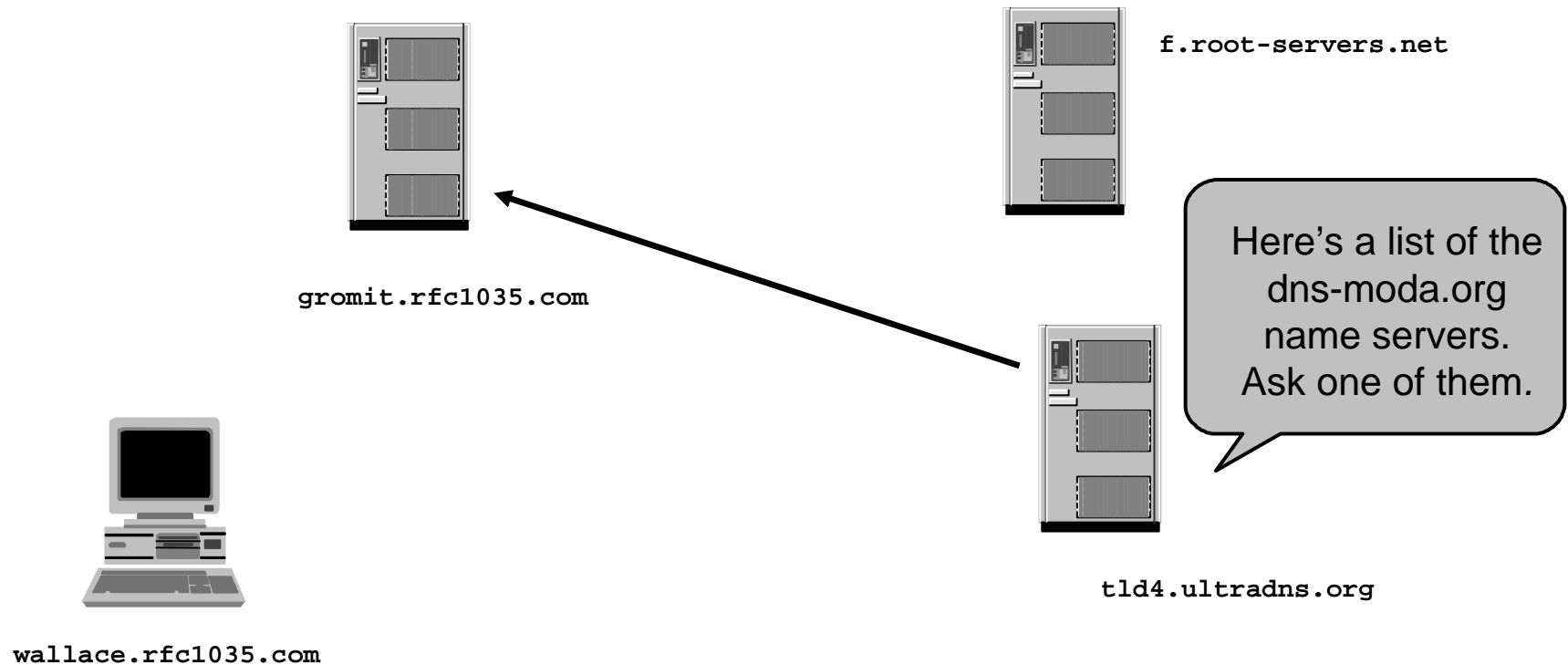
DNS Resolution - 4

- `gromit` now queries one of the `org` name servers, `tld4.ultradns.org`, for the IP address of `www.dns-moda.org`



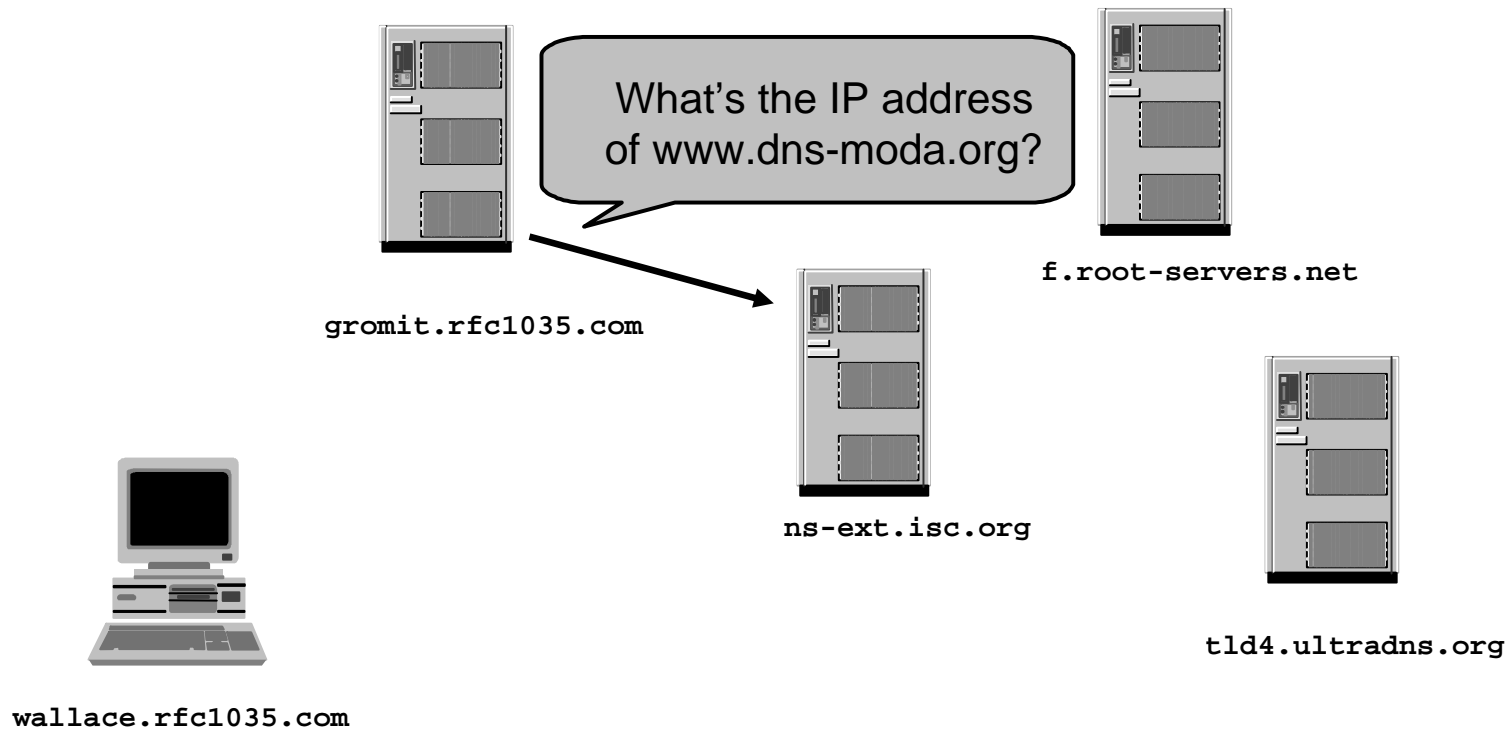
DNS Resolution - 5

- The org name server `tld4` refers `gromit` to the `dns-moda.org` name servers



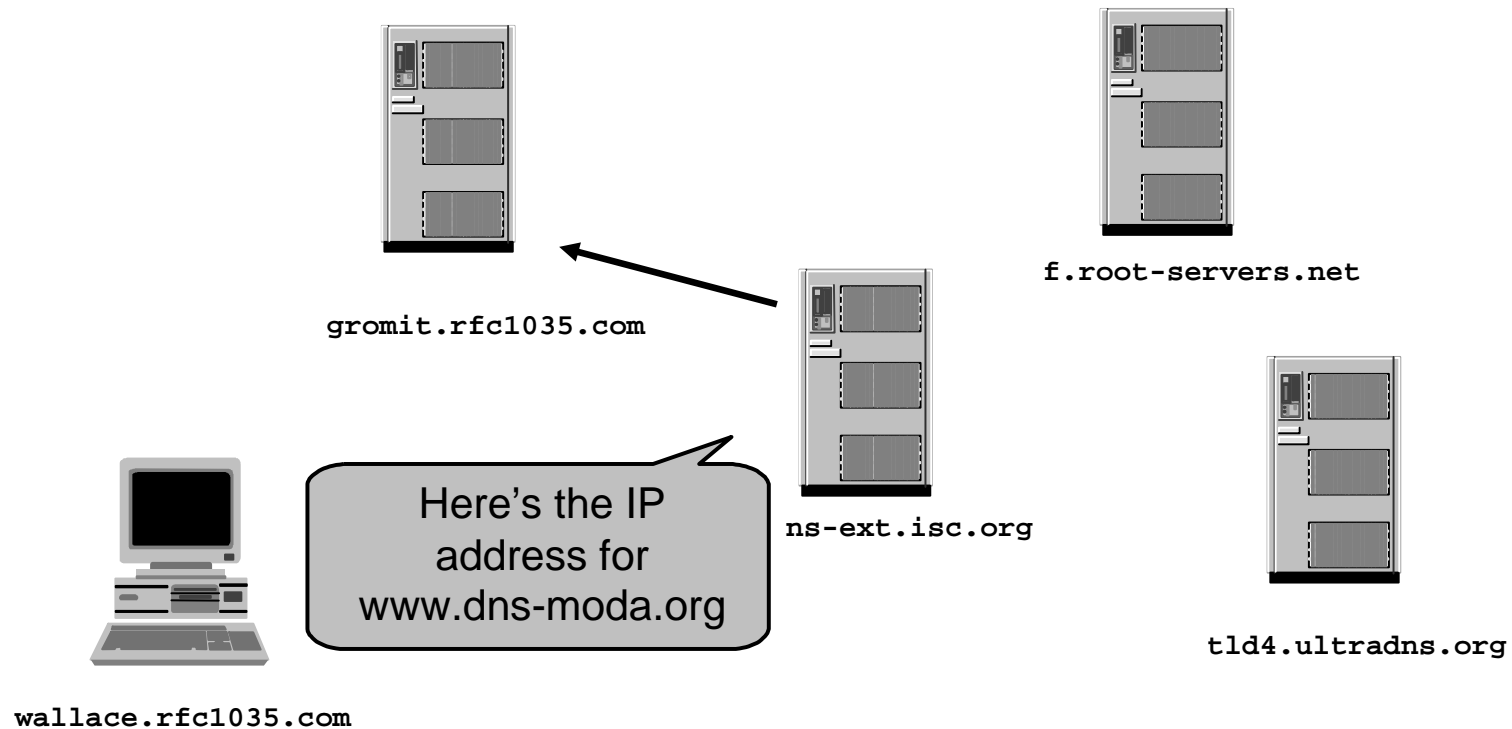
DNS Resolution - 6

- `gromit` now asks the `dns-moda.org` name server, `ns-ext.isc.org`, for `www.dns-moda.org`'s address



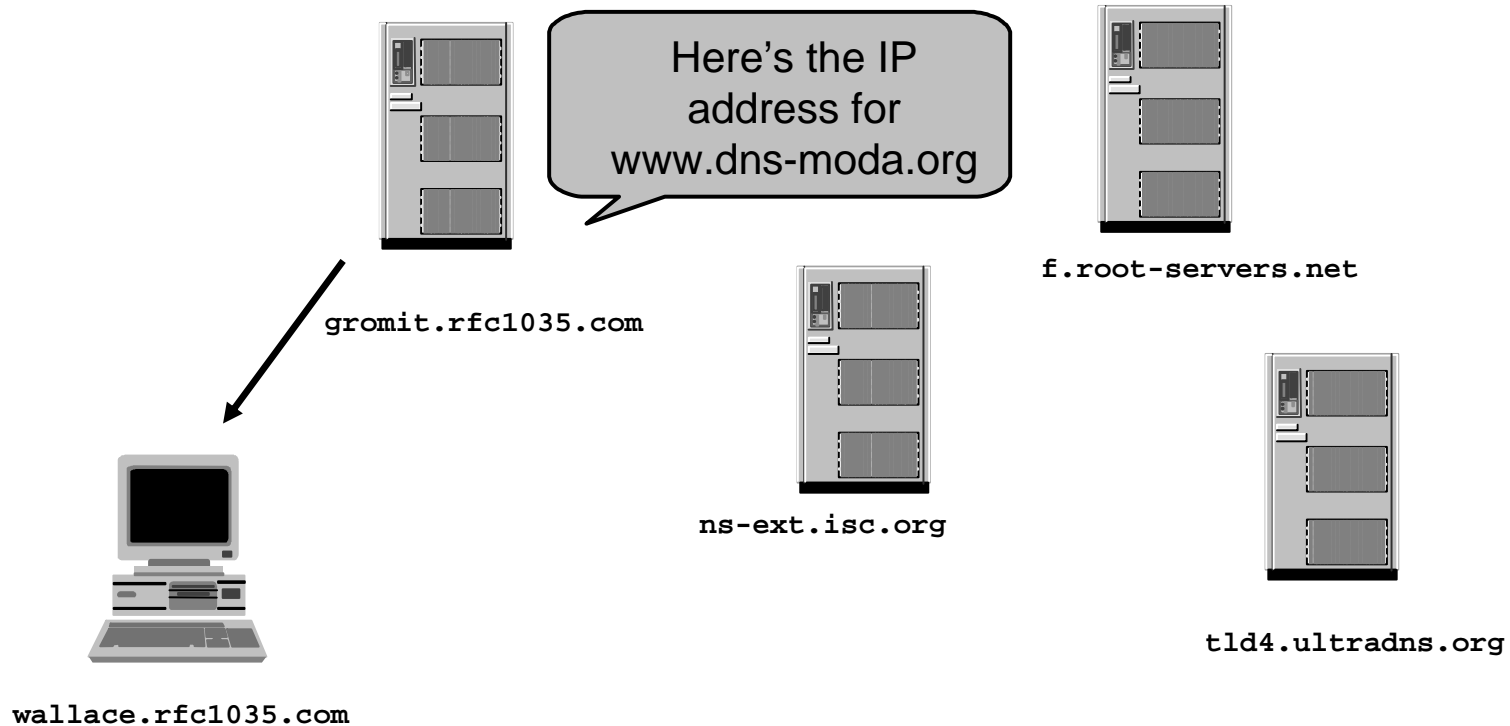
DNS Resolution - 7

- `ns-ext.isc.org` responds to `gromit` with `www.dns-modas.org`'s address



DNS Resolution - 8

- The name server `gromit` responds to `wallace` with `www.dns-moda.org`'s address



What's Wrong With That?

- Nothing: it all works fine.....
- BUT there's no authentication at all!
- A client can't tell:
 - Where an answer **really** came from
 - If the server that replied is telling the truth or not
 - If it received **exactly** what the server sent
 - This applies to the query `wallace.rfc1035.com` made and the lookups `gromit.rfc1035.com` performed to resolve that query

Attacking the DNS

- Bombard client with forged answers
 - Guess what the answer might be
- Intercept a response packet & modify it
 - Only works well if adjacent to client or server
- Set up a fake name server for some zone
 - Trick other name servers into querying the fake one
- Take control of the name server for some zone
 - Make it answer with false data
- Evil routing/peering tricks & hi-jack traffic
 - Inject bogus routes for the root servers (or the name servers for any other “interesting” zone)

What Does This Mean?

- A DNS client can't be sure of anything:
 - Did a lookup for `www.dns-moda.org` really get answered by the `dns-moda.org` name servers?
 - Did it get what a real `dns-moda.org` name server actually sent?
 - Is the name server that answered telling the truth?
- Did we get the true address of DNS-MODA's web server?
- Feel free to replace `dns-moda.org` with your favourite domain name....

Why Secure DNS?

- The DNS is not secure!!!
 - Servers could be lying
 - Cache poisoning attacks
 - Slave servers compromised
 - Name servers could be spoofed
 - Answers could be tampered with
 - UDP makes these attacks simple
- This is what Secure DNS is designed to solve
 - Attacks mentioned above can be detected

DNSSEC Overview

- Defined in RFCs 4033, 4034 & 4035
 - Obsoletes & cleans up a raft of earlier specifications, enhancements & extensions:
 - RFC2535, RFC3008, RFC3090, etc.
- Four new resource records:
 - DNSKEY, RRSIG, NSEC & DS
- Digital signatures of DNS data
- Industrial-strength cryptography:
 - RSA, DSA, Diffie-Hellman, SHA, MD5

What DNSSEC Does Not Do

- Prevent/thwart denial-of-service attacks
- Stop name server compromises
 - Buffer overflows
 - Run BIND9 to stop that!
 - Environment variable leakages
- Confidentiality of DNS data
 - The DNS is public after all...

What Secure DNS Proves

- Data integrity
 - Verify what was received was exactly what the name server sent
- Non-repudiation
 - Authenticate who/what signed the data
- Name server authenticity (in theory anyway)
 - An answer for **www.example.com** comes from the genuine name servers for **example.com**
 - Should be a chain of trust to the root

The Chain of Trust

- Public key for `dns-moda.org` is signed with the private key for `.org`
 - `.org` “trusts” the `dns-moda.org` key
- Public key for `.org` is signed with the private key for the root
 - Root zone “trusts” the `.org` key
- Everyone trusts the root zone’s public key
 - Openly published
 - Built in to every name server configuration?

Validation Model

- Answer for `www.dns-moda.org` is provably correct
 - It's been signed with the `dns-moda.org` key
 - So nobody could have tampered with the data
 - The `dns-moda.org` key was signed by the key for `.org` so the `dns-moda.org` key is OK
 - The `.org` key was signed by the root zone's key so the delegation to `org` can be trusted too
 - The root key is known and trusted by everyone

Technical Overview of DNSSEC

- A little more detail on what's involved with Secure DNS:
 - The cryptography-based technology
 - Keys and signatures
 - New resource records for DNSSEC

Public Key Cryptography

- Asymmetric encryption:
 - RSA, DSA
 - Rivest Shamir Adelman algorithm
 - Digital Signature Algorithm
 - Public key and private key pairs
 - Data encoded with public key can only be decoded with the corresponding private key and *vice versa*
 - Digital signatures

Cryptographic Hash Algorithms

- Often known as one-way hash algorithms
 - Can't reverse the hash function to work out what input produced a given hash value
- Typically a 64- or 128-bit hash of an input
 - No two inputs generate the same hash value
 - Collisions theoretically possible
- Example hash functions:
 - SHA (Secure Hash Algorithm)
 - MD5 (Message Digest 5)

Encryption Algorithms

- Implementations must support RSA & DSA
 - No RSA patent issues any more
 - DSA-SHA and RSA-MD5 are optional
- DSA is faster than RSA at signing
 - Takes longer to verify DSA signatures though
 - Verification should happen much more often than signing
 - Implies RSA is generally a better choice than DSA
- Using 2 or more algorithms doesn't provide stronger authenticity or "security"
 - DNS data will be insecure if any algorithm's key gets compromised

DNSSEC Signatures

- Don't explicitly sign the actual DNS data
 - Sign a hash of the data instead (SHA1)
 - Less data to sign
- Names must be normalised to a canonical form:
 - All in lower-case
 - Fully qualified domain names
 - Handled automatically by the zone signing tool

Verifying DNSSEC Signatures

- Signed response for a lookup returns the RRset and its corresponding signature (RRSIG)
 - Client computes hash value for the RRset
 - Applies public key to the RRSIG to get the hash value that was signed with the private key
 - If both hash values match, all is well
 - If not....

Creating DNSSEC Signatures

- Signing is generally performed when a new copy of the zone file is generated:
 - `dnssec-signzone` tool in BIND9
- Compute-intensive
 - Constructing hash values
 - Signing them with the private key
- Best if name servers don't generate signatures dynamically in response to each inbound query
 - Obvious Denial of Service attack
 - Also means keeping private key on-line

New DNS Record Types - 1

- DNSKEY
 - Public key of some sort
- RRSIG
 - Signature for some RRset in the zone
 - Includes *valid from* and *valid to* dates
 - Helps with the introduction of new keys
 - Crude revocation method

New DNS Record Types - 2

- NSEC
 - Name of next RRset in the zone
 - Also says what resource record types exist for some name
 - Needed for proof of non-existence:
 - *"The name you looked up does not exist"*
 - *"The name you looked up does not exist as an MX record"*
 - Last NSEC record points back to start of the zone
- DS
 - Meta delegation record in parent zone
 - Indicates child zone is DNSSEC-aware
 - Also says which child zone key(s) is valid

Creating DNSSEC Resource Records

- DNSKEY, RRSIG, NSEC & DS records are far too complex to be created by hand
 - Probably beyond the capabilities of any TLD registry's existing tools for producing zone files
- Tools are provided with BIND9
 - `dnssec-keygen`: generates keys
 - `dnssec-signzone`: signs a zone

Keys in DNSSEC Zones

- A signed zone will usually have two keys
 - A zone signing key (ZSK) which signs the resource records in the zone
 - A key signing key (KSK) which signs the ZSK
- DS record in parent zone identifies the KSK

Why the Key Signing Key?

- Key signing key simplifies DNSSEC deployment and administration:
 - Child zone can introduce a new zone signing key at any time
 - Doesn't have to interact with parent (e.g. TLD)
 - KSK can sign any new ZSK
- Child **MUST** inform parent when new KSK is introduced
 - Should be much less frequent than changing the zone's ZSK

Secure Dynamic Update

- Defined in RFC3007
 - But not well explained in BIND9 documentation yet
- On-line signing
 - BIND9 computes RRSIG and NSEC records on the fly
 - Dynamic update requests on signed zones
 - Potential for known plaintext attacks
- Name server must read the file containing the private key
 - Storing private keys on-line is probably a very bad idea

The Chain of Trust Revisited

- Validation involves checking parent zone's DS record and its RRSIG for a DNSSEC-aware child zone's KSK
 - KSK signs the ZSK which signs the zone data
- Process iterates to the root zone
 - In theory anyway....
- Root zone would have a DS record for .org's KSK which signs the ZSK .org uses for the RRSIGs over the DS records for its child zones
 - And so on....
- Public KSK for root zone published everywhere:
 - Hard-coded into name server configurations

DNSSEC-aware queries

- Use EDNS0 protocol (RFC2671)
 - Bigger DNS payloads/buffers
 - Standard DNS query only has 512 byte payload
 - Prevents truncated responses and TCP retries
 - The DO (DNSSEC OK) header bit
 - Indicates client understands DNSSEC
- DNSSEC-aware answer is **much** bigger
 - All the crypto stuff: RRSIGs, DNSKEYs, etc.
 - Exceeds conventional 512-byte limit

TLD Registry Considerations

- Resources
- Software
- Slave server arrangements
- Registry operations
- Registrar operations
- Signing policies
- Legal issues
- Trust anchors

Resource Considerations

- Signed zones are much bigger than their unsigned equivalents:
 - Approximately 4 times as many resource records
 - Each RRset gets an RRSIG plus an NSEC which also gets an RRSIG
 - Signed zone is typically 10 times bigger
 - RRSIG records are usually much larger than the RRset they sign
- More CPU power needed to sign the zone
 - Can use crypto hardware
 - Rely on Moore's law
- Signed responses are bigger
 - More bandwidth?
- Router/switch/firewall support for EDNS0

CPU Usage

- Illustrative zone signing times:
 - 2.5GHz PowerPC, 2 GB RAM, 1024-bit RSA key

Zone Size (# RRs)	Signing Time
1,000	4.5 seconds
5,000	20.5 seconds
10,000	41 seconds
50,000	205 seconds

- Zone signing is inherently parallelisable

Software Considerations

- Obviously need to run DNS software that supports current DNSSEC protocol
- Authoritative Name Servers:
 - Latest version of BIND9, 9.3.1
 - NLnetLabs NSD 2.3.1
 - Nominum ANS
- Resolving Name Servers:
 - Latest version of BIND9, 9.3.1
 - Nominum CNS

Slave Server Issues

- Slave (secondary) servers for zone have to support DNSSEC too
 - Don't need zone's private keys though
- Must also be willing/able to enable DNSSEC
- Resource considerations
 - RAM for signed zone
 - Bandwidth for queries & zone transfers
 - Propagation times for zone transfers
- Changes to Service Level Agreements?

Registry Operations

- Updates to back-end database
 - Support keying material (KSKs) for delegations
- Extensions to external interfaces
 - Web pages, EPP Schemas, email templates, etc.
- Allow for zone signing time latency when generating the TLD zone file
 - Extra hardware?
- Changes to processes and procedures
- Drills for emergency re-signing

Registrar Considerations

- Support for new registry/registrar interface(s)
 - Secure path between registry & registrar
- Code of conduct to protect against bad behaviour by registrar or registrant?
 - Registrar forges or spoofs the key for a delegation
- Education & outreach to customers
 - May need assistance from registry
- Business opportunities for new services
 - Offering key management facilities
 - Selling certificates

Choosing Key Lengths

- Key-signing keys should be no bigger than parent zone's key
 - Probably no point making them larger
 - Parent's key “strength” defines child's “strength”
- Use larger key sizes for long-lived RRSIGs
 - Beware of cryptanalysis
- Shorter key lengths OK for short-lived signatures
 - Typically valid for less than a week
- Consult crypto experts on time/size trade-offs

Good Crypto Policy

- Don't use same key for everything
- Maybe:
 - 1024-bit keys for signing zone data
 - 2048-bit keys for signing zone keys
- Change the keys “often enough”
- Perhaps consider alternate algorithms?
 - RSA is generally preferred over DSA
- Get advice from expert cryptographers

Legal Concerns

- What local laws apply to the use of cryptography and digital signatures?
 - Register keys with government authorities?
 - Patent & licensing issues?
- Does signing the zone imply changes to existing framework and contracts?
 - Who is authorised to insert keys in child zones?
 - Implicit contractual relationship?
- Liability issues if/when DNSSEC fails?
- Data protection & privacy legislation?

Trust Anchors

- The root zone is not signed. Yet....
- TLD's public key needs to be embedded in name server configurations
 - `trusted-keys{ }` statement in BIND
- How to get a new public key introduced to those configurations?

DNSSEC Problems - 1

- Bigger DNS packets
 - Typically break 512-byte payload limit
 - Need EDNS0 to allow bigger packets and signed responses
 - Idiot firewalls don't understand EDNS0 packets
- Zone files are much bigger and unreadable
- Signed zones can't be altered by hand
- Signing means changes to admin procedures:
 - *Check-out, modify, test, check-in, **sign** zone*
 - *Add/remove/change keys*

DNSSEC Problems - 2

- Parent zone signs child's key signing keys
- Implies closer coupling between parent and child zones
 - No bad thing, but **too** many broken/lame delegations
 - ~25% in tightly controlled registries
 - ??% in **.com**
 - High levels of DNS cluelessness

DNSSEC Problems - 3

- Root zone key is a weakness
 - Obvious single point of failure
 - Compromise of the key is potentially devastating
 - Reboot the DNSSEC-aware internet?
 - Will need to be changed regularly
 - Prevent cryptanalysis
 - How to get secure resolvers to pick up new key?
 - Likely to be a target for attack
 - How to propagate a new key after a compromise
 - Who controls the key?

DNSSEC Problems - 4

- Only 1 top-level domain is signed so far:
 - Sweden (`.se`) is not yet signing delegations
- Infrastructure domains might be first to see significant DNSSEC deployment
 - `in-addr.arpa`, `ip6.arpa`, `e164.arpa`
- DNSSEC knowledge and installed base is very low
- Better tools and procedures are needed:
 - Zone file maintenance
 - Key management
 - Key rollover (more on this later)

DNSSEC Problems - 5

- Added complexity in registry/registrar relationships
 - Make sure ZSKs are transmitted correctly
 - Secure path between registrant and registrar
 - Secure path between registrar and registry
- NSEC records allow the whole zone to be traversed
 - Political/legal concerns for some ccTLDs
 - Compilation copyright
 - Data protection
- Key rollover is awkward
 - Switching to a new key
 - New and old keys co-exist for a while
 - Both will sign the zone

DNSSEC Problems - 6

- DNS corner cases cause complexity
 - Wildcards
- No resolver API yet
 - What should this look like?
- Migration/roll-out scenarios
 - Is it OK to return unsigned responses to DNSSEC-aware clients?
 - Is it OK to return DNS data to DNSSEC-aware clients when/if validation fails?
 - Is known bad data better than nothing?

Key Rollover - 1

- Keys should be changed regularly:
 - Monthly? Weekly? Daily?
- Zone administrator can introduce new key alongside current key
 - RRsets can be signed by multiple keys
 - RRSIGs generated for each key in use
 - Get parent to generate a DS record for a new KSK
 - New ZSK's just need to be signed by the zone's KSK
- RRSIG record “valid from/to” timestamps help
 - New keys and RRSIGs introduced in advance
 - Period of dual-running

Key Rollover - 2

- When a parent's key changes, it has to re-sign the DS records of its secure child zones
 - No impact on delegated zones
 - Child doesn't need to change its ZSK or KSK
 - New keys at the parent won't affect child zones (i.e. TLD delegations)
- Proposal to use DNSKEY record for automatic DS creation for new zone signing keys
 - Greatly simplifies parent/child interaction
 - Less overhead for parent (e.g. a TLD registry)
 - New ZSK could be retrieved from the DNS
 - Might not be necessary for child to submit new ZSK via registrar

The Root Zone Key - 1

- Integrity of root key is critical
 - Compromise cannot be allowed (or suspected)
 - Break it and reboot the DNSSEC-aware internet
 - Obvious magnet for attackers
 - Massive single point of failure
- Root key(s) must change from time to time
 - Prevent cryptanalysis
- How will a new key for the root zone get propagated?
 - What about name servers and resolvers that can't/won't be reconfigured?

The Root Zone Key - 2

- Many political problems here
 - Any changes to the DNS root are sensitive
- Obvious concern:
 - Who will hold the root zone key(s) and sign the zone?
- Various IETF proposals to alleviate these problems:
 - Shared/multiple keys, threshold validation
- Expect a **lot** of debate:
 - ICANN, WSIS, IETF, ITU, etc.

DNSSEC at IETF

- Protocol development continues:
 - May mean a third iteration of the protocol
 - New & revived requirements
 - Prevent zone enumeration
 - Only sign DNSSEC-aware delegations
- Unclear when this work will be completed
- Concerns over future deployment
 - Backwards compatibility
 - Interoperability with current protocol
 - Possible "flag day"
 - Signalling of supported DNSSEC Protocol versions

Zone Enumeration

- Some TLDs are concerned about NSEC walking:
 - Just follow the NSEC chain to get a complete copy of the zone
 - Tools for this are now available
 - Extra query load on TLD name servers?
 - Showstopper for deployment in some TLDs
- IETF DNSEXT Working Group now considering a third iteration of the DNSSEC protocol
 - DNSSEC-ter should fix the zone enumeration problem
 - Work has been under way for about a year
 - Too early to predict outcomes, timelines, etc.

Opt-In

- Controversial
- Only sign DNSSEC-aware delegations
 - NSECv2 record points at next secure name in the zone, skipping over any unsigned names and delegations
- Could make signing very big zones easier
 - Probably 95% of `.com` doesn't want/care to be signed, so why bother signing it?
- May have security implications
 - Can't prove a name doesn't exist
 - It may exist but just isn't signed.....

Last Mile Issues - 1

- DNSSEC protocol is "done"
 - Much work remains however
- Applications support for DNSSEC
 - Need an API
 - Plug-ins for web browsers?
 - Killer app still to emerge
- Chicken-and-egg problem
 - No installed base because of low deployment
 - Negligible deployment as there's low perceived demand

Last Mile Issues - 2

- Current DNSSEC tools are primitive
 - Better signing & verification software
 - Troubleshooting
 - Identify where the chain of trust breaks
 - Key management & maintenance
 - Reminders to change keys & re-sign the zone
 - Parent checks on delegations?
 - Child checks on parent?

Last Mile Issues - 3

- The Internet has become a critical national & international resource:
 - e-commerce, email, instant messaging, telephony
- Governments beginning to look at critical internet infrastructure (i.e. DNS) as a matter of national security
 - Terrorism, disaster recovery, intrusion prevention
 - Government compulsion to deploy DNSSEC?
 - DNSSEC capabilities in government IT procurements?

DNSSEC Deployment Options

- Essentially three possibilities:
 - Early adopter
 - Testbed
 - Wait and see approach

Early Adopter

- Early adopter
 - Demonstrate TLD's technical expertise & core competence
 - Promote Secure DNS to local user base
 - Show TLD registry is serious about DNS security
 - Be proactive rather than reactive
- Possible disadvantages
 - Usual teething problems for being at the leading edge

Deployment Testbed

- Set up a testbed
- Study impact of DNSSEC deployment
 - Software, back-end database, hardware, network, etc.
 - Processes and procedures
 - Gain experience and expertise
- Outreach to local community
 - Government, industry, users
 - Gauge interest & demand
 - Education
- Dutch registry's **n1.nl**

Wait and see

- Wait and see approach
 - Uncertainties about DNSSEC-ter at IETF
 - Who knows when it will be ready?
 - Migration and backwards compatibility concerns
 - Demand from local community
 - Lack of applications?
 - Learn from experience of the first to deploy
- Disadvantages
 - May be forced into deployment by external circumstances

DNSSEC Deployment Strategy

- NIC-SE's stepwise approach is sensible and pragmatic:
 1. Just sign the TLD zone
 - Identify and fix any operational problems
 - Slave server issues, signing overheads, etc.
 2. Sign delegations for clueful registrants
 - Understand database and registrar interface issues
 - Refine processes and procedures
 3. Publish **.se** public key
 4. Encourage wider uptake of Secure DNS

Potential DNSSEC Applications

- DNS as a generic PKI?
 - DNS is ubiquitous and works!
 - DNSSEC means answers can be validated
- Validated data from DNS means it could be used to store and distribute crypto keys and certificates
 - Fetching keys becomes a (Secure) DNS lookup
 - IPsec, SSL, SSH keys
 - PGP & GPG keys?
 - X.509 Certificates

QUESTIONS?