

A Security Analysis of the Composition of ChaCha20 and Poly1305

Gordon Procter

Information Security Group, Royal Holloway,
University of London, London, UK
gordon.procter.2011@live.rhul.ac.uk

Abstract. This note contains a security reduction to demonstrate that Langley’s composition of Bernstein’s ChaCha20 and Poly1305, as proposed for use in IETF protocols, is a secure authenticated encryption scheme. The reduction assumes that ChaCha20 is a PRF, that Poly1305 is ϵ -almost- Δ -universal, and that the adversary is nonce respecting.

1 Introduction

There has recently been a proposal to the CFRG to consider a combination of ChaCha20 and Poly1305 for inclusion in future IETF protocols [11]. This proposal has come about, in part, due to concern over the reliance of existing IETF protocols on AES and the risk that advances in the cryptanalysis of AES could leave users without a good choice for a symmetric cryptographic primitive. A similar concern led to the SHA-3 competition; improvements to attacks against SHA-1 led NIST to transition to the SHA-2 family of hash functions and to announce the SHA-3 competition, with the aim of choosing an alternative hash function to ‘improve the robustness of NIST’s overall hash algorithm toolkit’ [12].

ChaCha20 is a stream cipher designed by Bernstein [4], based on the Salsa family of stream ciphers [5]; in both of these ciphers, the keystream is generated using a ‘block function’ in a mode reminiscent of the counter mode of operation for block ciphers. This note will assume that the ChaCha20 block function is a PRF with signature $CC : \{0, 1\}^{256} \times \{0, 1\}^{128} \rightarrow \{0, 1\}^{512}$, that is, 32-byte keys, 16-byte input, and 64-byte output. A few papers have attacked reduced round versions for ChaCha20 (for example, [1,9,14]) however this analysis has not contradicted the PRF security of ChaCha20. Poly1305 is a universal hash function [7] based on polynomial evaluation [6,8,16], also designed by Bernstein [3].

This note gives a reduction from the security of the proposed combination of ChaCha20 and Poly1305 to the PRF-security of the ChaCha20 block function. Although both primitives are believed to be secure, it is possible to combine secure primitives in a way that is insecure (for a discussion and formal treatment of this issue, see [2]); for this reason, a security analysis of the combined scheme is important. Note that the generic results of Bellare and Namprempre [2] do not apply in this case as those results assume that independent keys are used in the component primitives. The analysis presented in this note does not concern the assumption that ChaCha20 behaves as a PRF. The proof will also make use of the fact that Poly1305 is an ϵ -almost- Δ -universal (ϵ -A Δ U) hash function [15] where Δ represents addition modulo 2^{128} , which Bernstein [3] has shown to be the case for $\epsilon = \frac{8\lceil L/16 \rceil}{2^{106}}$ for messages of length at most L bytes.

Notation For brevity, the ChaCha20 block function will be denoted by CC , Poly1305 by $Poly$, and the composition as defined in draft-irtf-cfrg-chacha20-poly1305-00 [11] by $CC\&Poly$. The key will be denoted by k and the per-message nonce by N . Associated data, plaintext, and ciphertext will be denoted by A , P , and C respectively. The concatenation of strings x and y will be denoted by $x||y$ and $\text{trunc}_n(x)$ will represent the truncation of a string x to n bytes. The byte length of a string x will be denoted len_x ; in the computation of $\text{Poly}_r(A, C)$, $\text{len}_{A,C}$ will represent the 16-byte block consisting of a 8-byte representation of the length of A in bytes, concatenated with a 8-byte block corresponding to the length of C . The ChaCha20 block function outputs blocks of keystream that are 64 bytes wide, so the plaintext and ciphertext will be encrypted and

decrypted in blocks of 64 bytes. These blocks of plaintext and ciphertext will be denoted P_i and C_i with $P = P_1 || \dots || P_p$ and $|P_i| = 64$ bytes, except perhaps for P_p which may be shorter (similarly for C). When A and C are zero padded to fill 16-byte blocks, this will be denoted \bar{A} and \bar{C} . The notation $x \leftarrow_s X$ should be interpreted as the element x being uniformly sampled from the set X . Addition modulo 2^{128} will be denoted by $+$, with the corresponding subtraction operation denoted by $-$.

2 Specification

2.1 The Algorithms

ChaCha20 is a stream cipher proposed by Bernstein [4], which is designed following similar principles as Salsa20 [5] (an eSTREAM finalist). ChaCha20 generates a keystream by applying the ChaCha20 block function to the key, nonce, and a block counter. Plaintext is then encrypted using this keystream, with block i of the plaintext xored with the output of the ChaCha20 block function, evaluated using block counter i . Poly1305 is a polynomial-based universal hash function [6,7,8,16], also designed by Bernstein [3].

The syntax of the ChaCha20 block function and Poly1305 is given below:

$$\begin{aligned} \text{CC} &: \{0, 1\}^{256} \times \{0, 1\}^{32} \times \{0, 1\}^{96} \rightarrow \{0, 1\}^{512} \\ \text{Poly} &: \{0, 1\}^{128} \times \{0, 1\}^* \rightarrow \{0, 1\}^{128} \end{aligned}$$

That is, the ChaCha20 block function takes as input a 32-byte key, a 4-byte block number, and a 12-byte nonce, and outputs 64 pseudo-random bytes; Poly1305 takes as input a 16-byte key (with some specific bits set to zero) and a message of arbitrary length and outputs a 16-byte digest of the message. The output of Poly1305 is the truncation (to 16 bytes) of a polynomial evaluated in $\mathbb{F}_{2^{130}-5}$ at the key r (which first has certain bits ‘clamped’ to zero); the polynomial’s coefficients are determined by the message (each 16-byte message block is encoded to an integer modulo $2^{130} - 5$). When encrypted with a one-time pad, the digest output from Poly1305 forms an information-theoretic message authentication code (see [10] and [15]).

The internal details of these algorithms are not relevant to this note and readers are referred to papers such as [3] and [4] for further details.

2.2 The Composition

The composition defined in Section 2.8 of draft-irtf-cfrg-chacha20-poly1305-00 [11] has three main parts, informally given below and more precisely described by E_k^0 and D_k^0 of Figure 2:

key-derivation A one-time Poly1305 key, r , and pseudo-one-time-pad, s , are derived from k and N using the ChaCha20 block function and a block counter of 0.

encryption The plaintext is encrypted using ChaCha20, with block i of the plaintext xored with the output of ChaCha20 block function on input (k, i, N) .

tag generation Poly1305 is evaluated using the one time key r , which has certain bits set to zero. The input to Poly1305 is a message consisting of: associated data (padded with zeros to the next 16-byte block boundary), ciphertext (padded similarly), and a 16-byte block containing 8-byte representations of the length (in bytes) of both the associated data and the ciphertext. The pseudo-one-time-pad is then added (modulo 2^{128}) to the resulting digest.

3 Security Analysis

3.1 Security Model

The objective of this section is to demonstrate that the combination of ChaCha20 and Poly1305 described above is a secure authenticated encryption scheme. The accepted definition for a secure authenticated encryption scheme is one that provides both Indistinguishably under Chosen Plaintext Attacks (IND-CPA) and

the Integrity of Ciphertexts (INT-CTXT). These notions were introduced by Bellare and Namprepre [2] and together these these properties imply IND-CCA security. In fact, the stronger notion of IND\$-CPA security [13] can be shown to be achieved by this composition.

To model these two notions the adversary is given access to an encryption oracle and a decryption oracle and permitted to make at most q queries to these oracles. The proof proceeds via a series of games and the encryption and decryption oracles in Game i are denoted E^i and D^i respectively. An adversary breaks the IND\$-CPA security of the scheme if they can distinguish (C, T) generated by E^0 from a random bitstring of the same length (generated by an oracle denoted by $\$$); the adversary’s advantage against the IND\$-CPA security of a scheme is measured by $\mathbf{Adv}_{\text{ind\$-cpa}}^{\text{CC\&Poly}} = |\Pr[\mathcal{A}^{E^0} \rightarrow 1] - \Pr[\mathcal{A}^{\$} \rightarrow 1]|$. An adversary breaks the INT-CTXT security of a scheme if they can forge a ciphertext, i.e. output a tuple (N, A, C, T) with $D_k(N, A, C, T) = (N, A, P) \neq \perp$ where (N, A, C, T) is not the output from an encryption query and D outputs \perp to signify that the input was not a valid ciphertext. The advantage of this adversary is measured by $\mathbf{Adv}_{\text{int-ctxt}}^{\text{CC\&Poly}} = \Pr[\mathcal{A}^{E^0, D^0} \text{ forges}]$. A combined measure of the adversary’s advantage against both IND\$-CPA and INT-CTXT can be defined as $\mathbf{Adv}_{\text{ae}}^{\text{CC\&Poly}} = |\Pr[\mathcal{A}^{E^0, D^0} \rightarrow 1] - \Pr[\mathcal{A}^{\$, \perp} \rightarrow 1]|$, where \perp is an oracle that simply returns \perp (representing an invalid ciphertext) an all inputs.

The adversaries that are considered in this section will be restricted to nonce-respecting adversaries. This is a standard restriction for nonce-based authenticated encryption schemes and means that an adversary will never ask encryption queries (N, A, P) and (N, A', P') for $(A, P) \neq (A', P')$. There are no restrictions on the adversary’s use of nonces for decryption queries, however without loss of generality, it is assumed that an adversary makes no redundant queries; no query is repeated and the output from an E query is never input to the D oracle, or vice versa.

3.2 Reduction

It is assumed in this security analysis that no pair (k, N') is ever repeated, where N' is the 12-byte nonce that is input to the ChaCha20 block function; this assumption is critical to the security of CC&Poly. The draft recognises that not all protocols will use 12-byte nonces and ‘it is up to the protocol document to define how to transform the protocol nonce into a 12-byte nonce’ [11, Sect. 2.8]; one suggestion is that prepending a constant value could provide a way to expand a shorter nonce to 12 bytes.

If an implementation permits both 12-byte nonces and shorter nonces and an adversary is able to predict how a short nonce will be expanded to 12 bytes (for example, by guessing the value that will be prepended), then a nonce collision could be forced by querying the encryption oracle using a short N and a 12-byte N' which is the expanded version of N . In what follows, we will assume that all nonces are 12 bytes long and that no (key, nonce) pair is ever repeated to the encryption oracle; the protocol specification therefore must prevent nonce collisions of this form.

This section will assume that ChaCha20 is a PRF with signature $\text{CC} : \{0, 1\}^{256} \times \{0, 1\}^{128} \rightarrow \{0, 1\}^{512}$, that is, 32-byte keys, 16-byte input, and 64-byte output. This assumption has not been contradicted by any of the existing cryptanalysis of ChaCha20 and the analysis presented in this note does not concern this assumption. The proof will also make use of the fact that Poly1305 is an ϵ -almost- Δ -universal hash function.

The proof proceeds via a series of games, specified in Figures 1 and 2. Game 0 defines a combined IND-CPA and INT-CTXT game, with oracles that realise CC&Poly. The scheme specified in Game 4 clearly gives no adversary any advantage in either of the IND\$-CPA and INT-CTXT games: the ciphertext and tag are sampled independently of P and uniformly at random from $\{0, 1\}^{512}$ (as they would be if generated by $\$$) and it is impossible for an adversary to query D_4 with anything returning $(N, A, P) \neq \perp$.

The transitions between these games are justified as follows:

Games 0 and 1 If an adversary is able to distinguish between these two games, then they can distinguish ChaCha20 from a function chosen uniformly at random from the set of all functions with domain $\{0, 1\}^{128}$ and range $\{0, 1\}^{512}$. However, we assume that ChaCha20 is a PRF so no adversary gains a significant advantage through the transition between these games.

Games 1 and 2 These games are identical, on the condition that the inputs to URF in Game 1 never repeat. The inputs to URF are all of the form $(i||N)$; for each query, N is constant, but i is never reused

and no two encryption queries use the same value for N , therefore the random variables in Games 1 and 2 are identically distributed.

Games 2 and 3 These games are identical unless an adversary submits (N, A, C, T) to their decryption oracle and D_1 returns $(N, A, P) \neq \perp$. However, for each query that an adversary makes, this happens with probability at most ϵ (because Poly is ϵ -A Δ U) (see [10] and [15] for a proof of this). By a standard hybrid argument, the probability that an adversary making at most q queries successfully forges is at most $q\epsilon$.

Games 3 and 4 The random variables in these games are sampled in different orders, however the joint distributions are identical and therefore these games are identical.

A standard game-hopping argument allows the probability $\Pr(\mathcal{A}^{G(i-1)} \rightarrow 1)$ to be bounded in terms of $\Pr(\mathcal{A}^{G^i} \rightarrow 1)$:

$$\begin{aligned} \Pr(\mathcal{A}^{G^0} \rightarrow 1) &\leq \Pr(\mathcal{A}^{G^1} \rightarrow 1) + \mathbf{Adv}_{prf}^{\text{CC}}(\mathcal{B}) \\ \Pr(\mathcal{A}^{G^1} \rightarrow 1) &= \Pr(\mathcal{A}^{G^2} \rightarrow 1) \\ \Pr(\mathcal{A}^{G^2} \rightarrow 1) &\leq \Pr(\mathcal{A}^{G^3} \rightarrow 1) + q\epsilon \\ \Pr(\mathcal{A}^{G^3} \rightarrow 1) &= \Pr(\mathcal{A}^{G^4} \rightarrow 1) = \Pr(\mathcal{A}^{\$, \perp} \rightarrow 1) \end{aligned}$$

Bernstein [3] demonstrates that Poly1305 is ϵ -almost- Δ -universal for $\epsilon = \frac{8\lceil L/16 \rceil}{2^{106}}$, where L denotes the maximum byte length of messages. For this construction L is the largest possible value of $16(\lceil \frac{\text{len}A}{16} \rceil + \lceil \frac{\text{len}C}{16} \rceil + 1)$, because the specification of CC&Poly pads A and C to 16-byte blocks and adds an extra 16 bytes of message denoting the length of additional data and ciphertext.

Therefore it can be concluded that for every adversary \mathcal{A} there is an adversary \mathcal{B} against the PRF security of the ChaCha20 block function such that:

$$\mathbf{Adv}_{AE}^{\text{CC\&Poly}}(\mathcal{A}) = \left| \Pr(\mathcal{A}^{G^0} \rightarrow 1) - \Pr(\mathcal{A}^{G^4} \rightarrow 1) \right| \leq \mathbf{Adv}_{prf}^{\text{CC}}(\mathcal{B}) + q \frac{8(\lceil L/16 \rceil)}{2^{106}}.$$

GAME i :
 $k \leftarrow_{\$} \{0, 1\}^n$
 $b \leftarrow \mathcal{A}^{E_k^i, D_k^i}$
return ($b = 1$)

Fig. 1. Games used to reduce the security of CC&Poly to the PRF security of CC. Oracles E^i and D^i are defined in Figure 2

References

1. Jean-Philippe Aumasson, Simon Fischer, Shahram Khazaei, Willi Meier, and Christian Rechberger. New Features of Latin Dances: Analysis of Salsa, ChaCha, and Rumba. In Kaisa Nyberg, editor, *Fast Software Encryption*, volume 5086 of *Lecture Notes in Computer Science*, pages 470–488. Springer Berlin Heidelberg, 2008.
2. Mihir Bellare and Chanathip Namprempre. Authenticated Encryption: Relations among Notions and Analysis of the Generic Composition Paradigm. In Tatsuaki Okamoto, editor, *Advances in Cryptology ASIACRYPT 2000*, volume 1976 of *Lecture Notes in Computer Science*, pages 531–545. Springer Berlin Heidelberg, 2000.
3. Daniel J. Bernstein. The Poly1305-AES Message-Authentication Code. In Henri Gilbert and Helena Handschuh, editors, *Fast Software Encryption*, volume 3557 of *Lecture Notes in Computer Science*, pages 32–49. Springer Berlin Heidelberg, 2005.

$\underline{E}_k^0(N, A, P):$

```

 $r||s \leftarrow \text{trunc}_{32}(\text{CC}_k(0||N))$ 
for  $i = 1, \dots, p-1$  do
   $Z_i \leftarrow \text{CC}_k(i||N)$ 
   $C_i \leftarrow Z_i \oplus P_i$ 
 $Z_p^* \leftarrow \text{trunc}_{\text{len}_{P_p}}(\text{CC}_k(p||N))$ 
 $C_p \leftarrow Z_p^* \oplus P_p$ 
 $T \leftarrow \text{Poly}_r(\overline{A}||\overline{C}||\text{len}_{A,C}) + s$ 
return  $(N, A, C, T)$ 

```

$\underline{E}_k^1(N, A, P):$

```

 $r||s \leftarrow \text{trunc}_{32}(\text{URF}_k(0||N))$ 
for  $i = 1, \dots, p$  do
   $Z_i \leftarrow \text{URF}_k(i||N)$ 
   $C_i \leftarrow Z_i \oplus P_i$ 
 $Z_p^* \leftarrow \text{trunc}_{\text{len}_{P_p}}(\text{URF}_k(p||N))$ 
 $C_p \leftarrow Z_p^* \oplus P_p$ 
 $T \leftarrow \text{Poly}_r(\overline{A}||\overline{C}||\text{len}_{A,C}) + s$ 
return  $(N, A, C, T)$ 

```

$\underline{E}_k^2(N, A, P):$

```

 $r||s \leftarrow_s \{0, 1\}^{256}$ 
for  $i = 1, \dots, p$  do
   $Z_i \leftarrow_s \{0, 1\}^{512}$ 
   $C_i \leftarrow Z_i \oplus P_i$ 
 $Z_p^* \leftarrow_s \{0, 1\}^{81\text{len}_{P_p}}$ 
 $C_p \leftarrow Z_p^* \oplus P_p$ 
 $T \leftarrow \text{Poly}_r(\overline{A}||\overline{C}||\text{len}_{A,C}) + s$ 
return  $(N, A, C, T)$ 

```

$\underline{E}_k^3(N, A, P):$

```

 $r||s \leftarrow_s \{0, 1\}^{256}$ 
for  $i = 1, \dots, p-1$  do
   $Z_i \leftarrow_s \{0, 1\}^{512}$ 
   $C_i \leftarrow Z_i \oplus P_i$ 
 $Z_p^* \leftarrow_s \{0, 1\}^{81\text{len}_{P_p}}$ 
 $C_p \leftarrow Z_p^* \oplus P_p$ 
 $T \leftarrow \text{Poly}_r(\overline{A}||\overline{C}||\text{len}_{A,C}) + s$ 
return  $(N, A, C, T)$ 

```

$\underline{E}_k^4(N, A, P):$

```

 $r||T \leftarrow_s \{0, 1\}^{256}$ 
for  $i = 1, \dots, p-1$  do
   $C_i \leftarrow_s \{0, 1\}^{512}$ 
   $Z_i \leftarrow C_i \oplus P_i$ 
 $C_p \leftarrow_s \{0, 1\}^{81\text{len}_{P_p}}$ 
 $Z_p^* \leftarrow P_p \oplus C_p$ 
 $s \leftarrow T - \text{Poly}_r(\overline{A}||\overline{C}||\text{len}_{A,C})$ 
return  $(N, A, C, T)$ 

```

$\underline{D}_k^0(N, A, C, T):$

```

 $r||s \leftarrow \text{trunc}_{32}(\text{CC}_k(0||N))$ 
 $T' \leftarrow \text{Poly}_r(\overline{A}||\overline{C}||\text{len}_{A,C}) + s$ 
if  $T \neq T'$  return  $\perp$ 
for  $i = 1, \dots, p-1$  do
   $Z_i \leftarrow \text{CC}_k(i||N)$ 
   $P_i \leftarrow Z_i \oplus C_i$ 
 $Z_p^* \leftarrow \text{trunc}_{\text{len}_{C_p}}(\text{CC}_k(p||N))$ 
 $P_p \leftarrow Z_p^* \oplus C_p$ 
return  $(N, A, P)$ 

```

$\underline{D}_k^1(N, A, C, T):$

```

 $r||s \leftarrow \text{trunc}_{32}(\text{URF}_k(0||N))$ 
 $T' \leftarrow \text{Poly}_r(\overline{A}||\overline{C}||\text{len}_{A,C}) + s$ 
if  $T \neq T'$  return  $\perp$ 
for  $i = 1, \dots, p-1$  do
   $Z_i \leftarrow \text{URF}_k(i||N)$ 
   $P_i \leftarrow Z_i \oplus C_i$ 
 $Z_p^* \leftarrow \text{trunc}_{\text{len}_{C_p}}(\text{URF}_k(p||N))$ 
 $P_p \leftarrow Z_p^* \oplus C_p$ 
return  $(N, A, P)$ 

```

$\underline{D}_k^2(N, A, C, T):$

```

 $r||s \leftarrow_s \{0, 1\}^{256}$ 
 $T' \leftarrow \text{Poly}_r(\overline{A}||\overline{C}||\text{len}_{A,C}) + s$ 
if  $T \neq T'$  return  $\perp$ 
for  $i = 1, \dots, p-1$  do
   $Z_i \leftarrow_s \{0, 1\}^{512}$ 
   $P_i \leftarrow Z_i \oplus C_i$ 
 $Z_p^* \leftarrow_s \{0, 1\}^{81\text{len}_{C_p}}$ 
 $P_p \leftarrow Z_p^* \oplus C_p$ 
return  $(N, A, P)$ 

```

$\underline{D}_k^3(N, A, C, T):$

```

return  $\perp$ 

```

$\underline{D}_k^4(N, A, C, T):$

```

return  $\perp$ 

```

Fig. 2. Oracles used in the games introduced in Figure 1.

4. Daniel J. Bernstein. ChaCha, a variant of Salsa20. <http://cr.yp.to/papers.html#chacha>, 2008. Document ID: 4027b5256e17b9796842e6d0f68b0b5e.
5. Daniel J. Bernstein. The Salsa20 family of stream ciphers. In Matthew Robshaw and Olivier Billet, editors, *New Stream Cipher Designs*, volume 4986 of *Lecture Notes in Computer Science*, pages 84–97. Springer Berlin Heidelberg, 2008.
6. Jürgen Bierbrauer, Thomas Johansson, Gregory Kabatianskii, and Ben Smeets. On Families of Hash Functions via Geometric Codes and Concatenation. In Douglas R. Stinson, editor, *Advances in Cryptology CRYPTO 93*, volume 773 of *Lecture Notes in Computer Science*, pages 331–342. Springer Berlin Heidelberg, 1994.
7. J. Lawrence Carter and Mark N. Wegman. Universal classes of hash functions (extended abstract). In *Proceedings of the Ninth Annual ACM Symposium on Theory of Computing, STOC '77*, pages 106–112, New York, NY, USA, 1977. ACM.
8. Bert den Boer. A simple and key-economical unconditional authentication scheme. *Journal of Computer Security*, 2:65–72, 1993.
9. Tsukasa Ishiguro. Modified version of latin dances revisited: New analytic results of Salsa20 and ChaCha. Cryptology ePrint Archive, Report 2012/065, 2012. <http://eprint.iacr.org/>.
10. Hugo Krawczyk. LFSR-based hashing and authentication. In Yvo G. Desmedt, editor, *Advances in Cryptology CRYPTO 94*, volume 839 of *Lecture Notes in Computer Science*, pages 129–139. Springer Berlin Heidelberg, 1994.
11. Y. Nir and A. Langley. ChaCha20 and Poly1305 for IETF protocols. <https://datatracker.ietf.org/doc/draft-irtf-cfrg-chacha20-poly1305/>, Jul 2014.
12. National Institute of Standards and Technology. Announcing request for candidate algorithm nominations for a new cryptographic hash algorithm (SHA3) family. *Federal Register*, 72(212):62212–62220, Nov 2007. http://csrc.nist.gov/groups/ST/hash/documents/FR_Notice_Nov07.pdf.
13. Phillip Rogaway, Mihir Bellare, and John Black. OCB: A block-cipher mode of operation for efficient authenticated encryption. *ACM Trans. Inf. Syst. Secur.*, 6(3):365–403, August 2003.
14. Zhenqing Shi, Bin Zhang, Dengguo Feng, and Wenling Wu. Improved key recovery attacks on reduced-round Salsa20 and ChaCha. In Taekyoung Kwon, Mun-Kyu Lee, and Daesung Kwon, editors, *Information Security and Cryptology ICISC 2012*, volume 7839 of *Lecture Notes in Computer Science*, pages 337–351. Springer Berlin Heidelberg, 2013.
15. D. R. Stinson. On the Connections Between Universal Hashing, Combinatorial Designs and Error-Correcting Codes. *Electronic Colloquium on Computational Complexity (ECCC)*, 2(52), 1995.
16. Richard Taylor. Near optimal unconditionally secure authentication. In Alfredo Santis, editor, *Advances in Cryptology EUROCRYPT'94*, volume 950 of *Lecture Notes in Computer Science*, pages 244–253. Springer Berlin Heidelberg, 1995.